

How to get a grade 5 in Edexcel GCSE Computer Science

A grade 5 is a strong pass. You get it by knowing the core facts of Topics 1 to 5 well enough to collect the short-answer marks, doing the standard binary and file size working reliably, and writing Python that works for the more straightforward tasks in Paper 2. You do not need every hard mark: you need to stop dropping the ones you can already get.

What a grade 5 asks of you

- Accurate recall of the core terms and processes: the parts of the CPU, the fetch–decode–execute cycle, main memory and secondary storage, LANs and WANs, common protocols, types of malware, lossy and lossless compression.
- Explain answers that give a reason, not just a statement, for the 2-mark questions.
- Reliable by-hand working: binary to denary and back, binary addition, binary to hexadecimal, and simple image and sound file sizes.
- Trace tables completed correctly for short algorithms with one loop.
- In Paper 2: short programs that work, using input, if/elif/else, while and for loops, and simple lists, and fixing the obvious errors in code you are given.
- A sensible attempt at every question, including Discuss answers with at least two or three developed points.

Secure these topics first

- Data (Topic 2): binary and hexadecimal conversions, binary addition, data units and file sizes. These marks are reliable once practised.
- Computers (Topic 3): the von Neumann architecture, what each CPU part does, the fetch–decode–execute cycle, main memory and secondary storage, and what an operating system and utility software do.
- Networks (Topic 4): LANs and WANs, wired and wireless connections, the common topologies and what the main protocols are for.
- Issues and impact (Topic 5): malware, social engineering and how to protect against them, environmental impact and personal data. These use everyday ideas, so the technical terms are what lift your answer.
- Computational thinking (Topic 1): trace tables, linear search, bubble sort, truth tables with AND, OR and NOT, and the three types of error.
- Programming (Topic 6): variables, input and output, selection, loops, and string and list basics.

Question types to master

- 1-mark State, Give, Name, Identify and Define questions: learn the exact terms from the cheatsheets.
- 2-mark Explain questions: a point plus a reason, linked to the scenario.
- Trace tables and completing an algorithm with gaps.
- Conversions and file size calculations with all the working shown.
- Paper 2 tasks where you fix errors in, or complete, a program you are given. These are usually the most accessible programming marks.
- The Discuss question: plan three points, cover each aspect named, and give both sides.

Programming practice for Paper 2

- Program in a Python IDE three or four times a week for 20 to 30 minutes. Reading code is not the same as writing it.
- Build a set of small programs you can write from memory: a validation loop for a number in a range, a total and mean of a list, the largest item in a list, counting items that match, a linear search, reading the lines of a file.
- Learn to read error messages: the line number, and what `SyntaxError`, `NameError`, `TypeError` and `IndexError` usually mean.
- Remember that `input()` always gives a string. Use `int()` or `float()` before comparing numbers or doing arithmetic.
- Stick to the Python in the PLS so nothing in the exam looks unfamiliar.

A 10-week plan

- Aim for 3 to 4 hours a week, plus a Daily 5 every day.
- Weeks 1 and 2: Topics 1 and 2 notes and worksheets; binary practice every day; Python input, if and loops.
- Weeks 3 and 4: Topics 3 and 4; Python strings and lists; your first custom papers on Easy and Medium.
- Weeks 5 and 6: Topic 5; Python files and simple functions; one timed Paper 1 mock.
- Weeks 7 and 8: your weakest subtopics from Progress; the Paper 2 practice set; one Discuss answer each week.
- Weeks 9 and 10: timed mocks of both papers; redo the questions you dropped marks on; flashcards and cheatsheets every day.

Using GCSE Paper Builder

- Notes: for each subtopic, read the notes, do the worksheet, then check it against the answers. Keep the cheatsheet for last-minute revision.
- Grade-by-grade skill lists: use the grade 4 and 5 lists as a checklist and tick off each skill when you can do it without help.
- Daily 5: five questions a day from your weakest topics. Keep the streak going; little and often works.
- Flashcards: definitions, CPU parts, protocols, types of malware and utility software. Spaced repetition brings each card back just before you forget it.
- Custom paper: pick the subtopics you have just revised, choose Easy or Medium and a short length, and mark it with the mark scheme. Move up to Hard when you score well.
- Mock exams: sit Paper 1 timed. The Paper 2 practice set covers the Topic 6 skills: type your answers into a Python IDE and run them as well.
- Progress: check your weakest subtopics every week and redo the questions you dropped marks on. Use the Whiteboard for binary working, trace tables and flowcharts.
- If programming is where you are stuck, 1-to-1 tutoring with Chhetri Academy (free 30-minute trial lesson) can get you moving quickly.

Timing in the exam

- Paper 1: about 1.2 minutes per mark, so check the marks for each question and the clock as you finish it. Don't spend more than a minute or two on any 1-mark part.
- Answer every part. A blank scores nothing; a sensible key term might score.
- Paper 2: do the fix-and-complete tasks carefully and quickly to bank those marks.
- For each longer Paper 2 task, get the inputs and the basic structure running first, then add the details one at a time.
- Save and run each file before you move on to the next question.

Traps at grade 5

- Vague answers such as 'it's faster' or 'it's more secure' without saying how or why.
- Mixing up bits and bytes, forgetting to divide by 8, or using 1000 instead of 1024 between KiB and MiB.
- Describing when the question says explain, or giving three answers when it asks for two.
- Confusing main memory (RAM) with secondary storage.
- In Paper 2: indentation errors, using = instead of == in a condition, and forgetting `int()` so numbers are compared as text.
- Saving a Paper 2 file under the wrong name, or not saving it at all.
- Giving up on the later Paper 2 tasks. A partial program with the inputs and a loop can still earn marks.

Free notes, worksheets, practice papers and mock exams at chhetriacademy.com. Written for this site; not taken from past papers.
Stuck? 1-to-1 tutoring with a free 30-minute trial lesson.