

Grade 8 checklist

42 skills. Tick each one when you can do it. For grade 8 you need the skills for the grades below it too.

Computational thinking

- ☐ **Convert between flowcharts, pseudocode and code** Turn a flowchart into Python, or Python into a flowchart, keeping exactly the same logic. (Flowcharts, pseudocode and program code)
- ☐ **Process arrays and records with loops** Loop through every item or record to total, count, search or output fields. (Variables, constants and data structures (strings, arrays and records))
- ☐ **Write complex conditions with brackets and NOT** Build conditions such as not $(x == 0 \text{ or } y == 0)$ that do exactly what is needed. (Arithmetic, relational and logical operators)
- ☐ **Trace nested loops and 2D arrays accurately** Keep track of both loop counters as the inner loop runs in full on each pass of the outer loop. (Tracing algorithms and trace tables)
- ☐ **Explain how to prevent runtime errors** E.g. check that a divisor is not zero, or that an index is in range, before using it. (Syntax, logic and runtime errors)
- ☐ **Write code for a linear search** Use a loop, a comparison and a flag that stops the search when the item is found. (Linear and binary search)
- ☐ **Write code for a bubble sort** Nested loops that compare adjacent items, swap them with a temporary variable and use a flag. (Bubble sort and merge sort)
- ☐ **Evaluate algorithms by time and memory** Judge which algorithm suits a scenario, using compares, passes and memory use. (Efficiency of algorithms)
- ☐ **Use a truth table to solve a problem** Find the input combinations that give a required output. (Truth tables and logic)

Data

- ☐ **Explain how one pattern has different meanings** The bits have no meaning on their own: the program using them decides whether they are a number, a character, a colour or an instruction. (Binary representation and unsigned binary–denary conversion)
- ☐ **Find the range for any number of bits** n bits store -2^{n-1} to $2^{n-1} - 1$, e.g. 4 bits store -8 to $+7$. (Two's complement)
- ☐ **Compare a pattern's unsigned and signed values** When the MSB is 1, the unsigned value is 256 more than the two's complement value, because that bit is worth $+128$ instead of -128 . (Two's complement)
- ☐ **Explain why arithmetic shifts suit signed numbers** Copying the sign bit keeps a negative number negative, so it is correctly divided by 2 for each place. (Logical and arithmetic shifts)
- ☐ **Explain that hex does not save memory** The computer still stores the value in binary; hex is only the way it is shown to people. (Hexadecimal)
- ☐ **Explain why '7' and 7 are stored differently** The character '7' is stored as its ASCII code, 55 (011 0111), not as the binary number 7 (000 0111). (Characters and ASCII)

- ☐ **Work backwards to find colour depth** Colour depth = total bits ÷ number of pixels, after converting the size into bits. (Bitmap images)
- ☐ **Calculate the minimum bits for a requirement** Find the smallest power of 2 that covers every value needed. (Limitations of binary representation)
- ☐ **Solve multi-step storage capacity problems** Number of files that fit = capacity ÷ size of one file, in the same unit, rounded down. (Units and file size calculations)
- ☐ **Evaluate compression methods for a scenario** Weigh file size and speed against quality and whether the data must be exact. (Lossless and lossy compression (incl. RLE))

Computers

- ☐ **Explain a drawback of one shared memory** Instructions and data travel on the same buses, so they cannot be fetched at the same time and the CPU may have to wait. (Stored program concept and von Neumann architecture)
- ☐ **Justify a storage choice for a scenario** Weigh the user's needs against the features of each type of storage and reach a reasoned conclusion. (Main memory and secondary storage)
- ☐ **Explain why SSDs should not be defragmented** Access time does not depend on where data is stored, and the extra writing wears out the flash memory. (Utility software)
- ☐ **Evaluate ways of making software robust** Weigh code reviews, audit trails and testing against their cost and the fact that none of them can find every problem. (Robust software: audit trails and code reviews)

Networks

- ☐ **Explain why the internet survives link failures** There are many routes and no central point, so routers send packets around a failed link or router. (The internet: IP addressing and routers)
- ☐ **Discuss wired versus wireless for a scenario** Weigh up both types for each group of users and reach a justified recommendation, often a mixture of both. (Wired and wireless connectivity and performance)
- ☐ **Explain which protocols a scenario uses** Name the protocol for each stage and justify it, e.g. HTTPS for a payment and IMAP for reading email on several devices. (Protocols)
- ☐ **Apply the model to an unfamiliar situation** For example, switching from Ethernet to Wi-Fi changes only the link layer, so the web browser is unaffected. (The TCP/IP model)
- ☐ **Recommend a topology for a scenario** Balance cost, performance, fault tolerance and ease of expansion, and justify the choice. (Network topologies)
- ☐ **Evaluate the limits of penetration testing** A test is a snapshot of one moment, so new vulnerabilities appear afterwards and testing must be repeated. (Network security and identifying vulnerabilities)
- ☐ **Evaluate a combination of protection methods** Explain why no single method is enough and how the three methods cover each other's gaps. (Protecting networks)

Issues and impact

- ☐ **Weigh up a technology's overall environmental impact** In a 6-mark answer, balance the harms against the benefits in the given context and reach a justified conclusion. (Environmental issues)

- ☐ **Discuss the benefits and risks of collecting data** Balance convenience and better services against privacy, consent and security, then reach a conclusion. (Personal data: privacy, ownership, consent and data protection)

- ☐ **Discuss using AI in a given context** Weigh benefits such as speed and consistency against safety, bias, accountability and liability, then conclude. (Artificial intelligence, machine learning and robotics)

- ☐ **Discuss open-source versus proprietary for a user** Weigh cost, income, support, customisation and control for the person or organisation, then recommend one. (Intellectual property and licensing)

- ☐ **Discuss the threats facing an organisation** Link each threat to the organisation's systems and data, and explain the likely impact. (Malware, social engineering and technical vulnerabilities)

- ☐ **Discuss how an organisation should protect data** Recommend layered measures that fit the organisation's data, threats and budget, with reasons. (Protecting digital systems and data (incl. encryption))

Problem solving with programming

- ☐ **Evaluate fitness for purpose and efficiency** Use test results and the number of comparisons, loop passes and memory used to judge a program. (Developing code: readability, errors, validation and testing)

- ☐ **Combine constructs to solve unfamiliar problems** Decompose the task, choose the right constructs and give each loop a single exit point. (Sequence, selection and iteration)

- ☐ **Store and process records in a list** Hold each record as an inner list of fields and loop through the records to search or total them. (Data types and data structures)

- ☐ **Code authentication using an ID lookup** Search the stored IDs for a match and check the password stored with that ID. (Input, output and file handling)

- ☐ **Evaluate complex Boolean expressions** Know that Python applies not first, then and, then or, and use brackets to control the order. (Operators)

- ☐ **Explain local and global variables** Say where each variable can be used and why local variables are better practice. (Subprograms and libraries)

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.