

Grade 7 checklist

52 skills. Tick each one when you can do it. For grade 7 you need the skills for the grades below it too.

Computer systems (Paper 1)

- ☐ **Explain each component's role during the cycle** Link each register and component to exactly what it stores, manages or controls at the fetch, decode and execute stages. (Architecture of the CPU)

- ☐ **Choose a CPU for a task and justify** Use the figures in a table and link each characteristic to the task, e.g. multitasking benefits from more cores. (CPU performance)

- ☐ **Describe how an embedded system controls a device** Describe the inputs (e.g. sensors), the decision it makes and the output it controls in an unfamiliar device. (Embedded systems)

- ☐ **Explain why virtual memory slows a computer** Secondary storage is much slower than RAM, and data must be moved back into RAM before the CPU can use it. (Primary storage (RAM, ROM, cache, virtual memory))

- ☐ **Evaluate storage options using given data** Use capacities, speeds and costs from a table, including checking by calculation whether the data will fit. (Secondary storage)

- ☐ **Solve multi-step capacity problems** E.g. find how many files fit in the space left on a device, rounding down to a whole number of files. (Units and capacity)

- ☐ **Reason about ranges and bit patterns** E.g. n bits give 2^n values, odd numbers end in 1, and why each hex digit matches exactly four bits. (Binary, denary and hexadecimal)

- ☐ **Explain the effect of bits being lost** 1s shifted off the right make the result round down; 1s shifted off the left make the result wrong (overflow). (Binary addition, overflow and shifts)

- ☐ **Compare ASCII and Unicode, including file size** Unicode covers many more characters but uses more bits per character, so text files are larger. (Characters: ASCII and Unicode)

- ☐ **Calculate an image's file size** Colour depth $\times$ width $\times$ height gives the size in bits; divide by 8 for bytes. (Images)

- ☐ **Calculate the file size of a recording** Sample rate $\times$ bit depth $\times$ duration in seconds gives bits; divide by 8 for bytes. (Sound)

- ☐ **Explain advantages and disadvantages of each type** Lossy gives much smaller files but loses quality; lossless keeps every bit but saves less space. (Compression)

- ☐ **Use data to identify the limiting factor** Decide from measurements whether bandwidth, latency or errors are causing poor performance, and justify a matching improvement. (LANs, WANs and network performance)

- ☐ **Recommend and justify a network model** Weigh cost, number of computers, security and backups for the organisation, then give a justified recommendation. (Client-server and peer-to-peer)

- ☐ **Choose and justify hardware for a scenario** Identify each item needed, e.g. for a café with Wi-Fi and a wired till, and explain why each one is needed. (Network hardware)

- ☐ **Calculate the connections in a full mesh** A full mesh of n devices needs $n(n - 1) \div 2$ connections; a star needs n cables. (Star and mesh topologies)

-
- ☐ **Recommend a topology for a scenario** Weigh reliability against cost for the organisation, and justify your choice. (Star and mesh topologies)
-
- ☐ **Recommend connections for a scenario** Choose Ethernet, Wi-Fi or Bluetooth for each device and justify the choice using the scenario. (Ethernet, Wi-Fi and Bluetooth)
-
- ☐ **Explain why longer keys are more secure** Each extra bit doubles the number of possible keys, so a brute-force attack takes far longer. (Encryption)
-
- ☐ **Explain why an IPv4 number cannot exceed 255** Each part is 8 bits, which gives $2^8 = 256$ values, from 0 to 255. (IP and MAC addressing)
-
- ☐ **Describe the roles of TCP and IP** TCP splits data into numbered packets, checks and reassembles them; IP addresses and routes the packets. (Standards and common protocols)
-
- ☐ **Apply layers to an unfamiliar scenario** E.g. explain why a browser still works when a laptop switches from Ethernet to Wi-Fi. (The concept of layers)
-
- ☐ **Explain the concept of SQL injection** SQL code typed into an input box is run by the database, e.g. to bypass a login or reveal data. (Threats to computer systems and networks)
-
- ☐ **Match prevention methods to specific threats** E.g. training for phishing, input validation for SQL injection, encryption for interception. (Identifying and preventing vulnerabilities)
-
- ☐ **Explain several OS functions in a scenario** E.g. explain how the OS lets a user stream music, write an essay and print at the same time. (Operating systems)
-
- ☐ **Explain why SSDs should not be defragmented** They have no moving parts, so there is no speed gain, and the extra writes shorten the SSD's life. (Utility software)
-
- ☐ **Explain how a law affects an organisation** Apply the principles of the Data Protection Act to how a school or business handles personal data. (Legislation)
-
- ☐ **Recommend a type of software for a scenario** Weigh cost, support, customisation and compatibility for the user, and justify the choice. (Open source and proprietary software)
-

Algorithms and programming (Paper 2)

- ☐ **Explain the benefits of decomposition** Smaller parts are easier to code and test, can be shared between programmers and can be reused. (Computational thinking)
-
- ☐ **Explain how structure diagrams help development** Each module can be coded and tested separately, shared between programmers and reused. (Inputs, processes, outputs and structure diagrams)
-
- ☐ **Convert between a flowchart and code** Turn a flowchart into code, or code into a flowchart, keeping the same logic. (Pseudocode and flowcharts)
-
- ☐ **State the purpose of an algorithm** Trace it with sample data and describe what it does overall, such as 'finds the largest value'. (Trace tables and common errors)
-
- ☐ **Complete or identify search algorithm code** Recognise each search from its code and fill in missing lines, such as `first = mid + 1`. (Linear and binary search)
-
- ☐ **Complete bubble sort code, including the swap** Use a temporary variable to swap two items, and a flag to record whether a swap happened. (Bubble sort)
-

-
- ☐ **Give advantages and disadvantages of merge sort** Much faster on large lists, but uses more memory and is harder to code. (Merge sort)
-
- ☐ **Explain when insertion sort is a good choice** It is quick for small lists and lists that are nearly sorted, and uses little memory. (Insertion sort)
-
- ☐ **Choose and write the most suitable loop** Use for when the number of repeats is known and do ... until when it must run at least once. (Sequence, selection and iteration)
-
- ☐ **Use MOD and DIV to solve problems** Test for even numbers, or split a total into units, such as minutes and seconds. (Arithmetic, comparison and Boolean operators)
-
- ☐ **Work out the result and type of casts** For example `str(15) + "0"` gives the string "150". (Data types and casting)
-
- ☐ **Loop through each character of a string** Use for `i = 0` to `word.length - 1` with `word.substring(i, 1)`. (String manipulation)
-
- ☐ **Process data read from a file** Cast each line to the right type, then total, count or search it. (File handling)
-
- ☐ **Use a 2D array** Access an element with two indexes, such as `grid[2, 3]`. (Arrays and records)
-
- ☐ **Combine conditions with AND and OR** For example `WHERE Genre = 'Sci-fi' AND Minutes > 130`. (SQL)
-
- ☐ **Write a function with parameters** Use function ... endfunction, with parameters and a return statement. (Functions and procedures)
-
- ☐ **Explain local and global variables** A local variable exists only in its sub program; a global variable can be used anywhere. (Functions and procedures)
-
- ☐ **Pick a random element from an array** For an array of `n` items use the index `random(0, n - 1)`. (Random number generation)
-
- ☐ **Explain how to anticipate misuse** Plan for unexpected input, repeated login attempts and people trying to break the program. (Defensive design)
-
- ☐ **Use test results to find and fix errors** Compare expected and actual results to locate a logic error and correct the line. (Testing)
-
- ☐ **Complete a three-input truth table** List all 8 input combinations in order and work out each output. (Logic gates, diagrams and truth tables)
-
- ☐ **Choose a translator for a scenario** Interpreter for development and debugging; compiler for distributing a finished program. (Languages and translators)
-
- ☐ **Describe using debugging tools to find errors** Use breakpoints, stepping and watching variables to find a logic error. (The Integrated Development Environment)
-
- ☐ **Validate input inside a loop** Keep asking until the input meets the rule, before using it. (Paper 2 Section B practice: design, write, test and refine)
-

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.