IQM3010 A1 Teacher reference commentary

IQM3010 Computer Science · Assessment 1 written examination · Sample 2026

STAFF STANDARDISATION MATERIAL · REVIEW BEFORE USE

Selected fictional examples for teacher standardisation. Consult this proposed commentary only after independent marking. Partial extracts cannot establish a whole-script judgement, and all proposed marks remain subject to human review.

Source binding. Prepared against pack revision 0.3. Recheck the hashes below before use if the assessment is amended.

Controlled source	SHA-256
Candidate question paper	a5ef3ea246c223 976a813cd2e86f0df502d79bde74e034 157a3183da8eaf674d
Candidate answer booklet	d0073c92c3d144b9e5c8a36 285fb44c0b3a53d8cb00cab97 965e1df82a083fe4
Staff mark scheme and AO1 map	4eb25cea6f268 803b2eaebd8a16 329 773c532e5de35 779 703c8e8ea8d7da1e72

Selected task B1

Processor roles, the fetch stage with registers and buses, volatile/non-volatile storage, an input device and two operating-system roles.

Example X

(a) The program counter stores the address of the instruction currently being executed. The control unit decodes instructions and coordinates their execution. The ALU carries out arithmetic calculations and logical comparisons.

(b) The PC address is copied to the MAR. The MAR sends that address along the address bus and a read signal travels on the control bus. The instruction is transferred from memory along the data bus to the MDR, then copied to the CIR. The PC is incremented so it holds the address for the following instruction.

(c) RAM is volatile, so it loses its contents when power is removed; secondary storage is non-volatile and retains stored data.

(d) A keyboard is an input device; it lets a user enter text and commands into the computer.

(e) The operating system manages memory and files, and provides an interface and services through which applications can use the computer's resources.

Reference judgement: 12/13. Proposed subpart judgements follow.

- **(a) 2/3 —** Credit: CU decodes and coordinates instruction execution; ALU performs arithmetic or logical operations. Not credited: PC holds the address of the next instruction.
- **(b) 5/5 —** Credit: PC address copied to MAR; Address sent on address bus and read signal on control bus; Fetched instruction travels via data bus to MDR; MDR contents copied to CIR; PC incremented/prepared for following instruction.
- **(c) 1/1 —** Credit: Contrasts RAM loss with secondary-storage retention when power is removed.
- **(d) 2/2 —** Credit: Names a valid input device; Accurately states the device function.
- **(e) 2/2 —** Credit: States one valid operating-system role; States a second valid operating-system role.

Example Y

(a) The PC stores the address of the next instruction. The CU stores the instruction being used. The ALU does calculations.

(b) The PC address is copied to the MAR. The MAR puts the address on the address bus. Memory sends the instruction on the data bus to the MDR. The MDR copies it to the accumulator. The PC then increases for the following instruction.

(c) RAM loses data when power is removed. Secondary storage also loses data unless the computer is shut down correctly.

(d) A keyboard is an input device; it sends the output to the screen.

(e) The OS manages files and runs applications.

Reference judgement: 8/13. Some CPU knowledge is secure, but the CU role is confused and the fetch sequence omits the control-bus read signal and MDR-to-CIR transfer. Device-function and OS examples are partly secure.

- **(a) 2/3 —** PC and ALU functions earn credit; the CU is described as storing rather than decoding/coordinating.
- **(b) 3/5 —** Correct PC-to-MAR, memory-to-MDR and PC increment points; no control-bus read signal or CIR transfer.
- **(c) 0/1 —** The secondary-storage contrast is incorrect; it retains data without power.
- **(d) 1/2 —** The device is valid but its stated function is not.
- **(e) 2/2 —** Both are acceptable operating-system roles.

Example Z

(a) The program counter keeps the address of the next instruction. The control unit stores data from memory. The ALU does addition and subtraction.

(b) The PC is copied to the MAR and the address is sent to memory. The instruction is then put in the CIR. The PC increases for the next instruction.

(c) RAM and an SSD both lose their contents when the power is removed.

(d) Keyboard.

(e) [No response]

Reference judgement: 5/13. Proposed subpart judgements follow.

- **(a) 2/3 —** PC and ALU roles are correct; the control unit does not store fetched data.
- **(b) 2/5 —** Earns PC-to-MAR and PC-increment points. Omits the address bus plus control-bus read signal, the data-bus transfer to MDR, and the MDR-to-CIR step.
- **(c) 0/1 —** The contrast is incorrect; secondary storage retains data without power.
- **(d) 1/2 —** Names a valid input device but gives no function.
- **(e) 0/2 —** No operating-system roles are stated.

Selected task C3

Time-growth comparison, space efficiency, tractability and why algorithms for the same problem can differ in efficiency.

Example X

(a) Algorithm X has lower time growth: it is linear, O(n), while Y is O(n^2). At n=10,000 the examples are 10,000 versus 100,000,000 comparisons, and as n grows n^2 rises faster than n.

(b) Space efficiency is the amount of memory or storage an algorithm requires.

(c) A problem is tractable when it can be solved within feasible time and memory limits as input grows. It is intractable when the required resources grow so much that solving larger inputs becomes impractical.

(d) Algorithms use different sequences of operations and data-handling choices, so their operation or storage growth can differ. They can still produce the same correct answer for the same problem.

Reference judgement: 9/10. Proposed subpart judgements follow.

- **(a) 3/3 —** Credit: Identifies X as lower time growth; Identifies X as linear O(n); Explains n versus n^2 using the stated comparison counts/growth.
- **(b) 1/2 —** Credit: Defines space efficiency as memory/storage required. Not credited: Relates memory growth to n/input size.
- **(c) 3/3 —** Credit: Defines tractable using feasible resources as size grows; Defines intractable using infeasible resource requirements as size grows; Mentions time or space limits.
- **(d) 2/2 —** Credit: Explains different processes can yield different growth; States both can solve the same problem correctly despite differing growth.

Example Y

(a) X has lower time growth and is O(n). As n increases, n grows more slowly than n squared, so X needs fewer comparisons.

(b) Space efficiency is how much memory an algorithm uses. It grows faster if the algorithm is faster.

(c) A tractable problem can be solved in a reasonable time. An intractable problem is just a difficult problem.

(d) Different algorithms use different steps, so they can perform different numbers of operations.

Reference judgement: 6/10. Time-growth comparison is strong, but the candidate only partly explains space scaling and tractability and does not state that both algorithms can solve the same problem correctly.

- **(a) 3/3 —** Identifies X, linear growth and the n versus n-squared comparison.
- **(b) 1/2 —** Defines the memory measure but does not relate storage growth to input size.
- **(c) 1/3 —** The tractable description earns credit; the intractable answer and resource-limit explanation are missing.
- **(d) 1/2 —** Explains how different processes can create different operation growth, but does not state that both can return the same correct result.

Example Z

(a) Algorithm X is lower because it makes fewer comparisons for the example. I think both are probably O(n^2).

(b) Space efficiency means how fast the program runs.

(c) [No response]

(d) [No response]

Reference judgement: 1/10. Proposed subpart judgements follow.

- **(a) 1/3 —** Identifies X as lower for the given comparison; gives the wrong growth order and does not correctly compare n with n-squared as size grows.
- **(b) 0/2 —** Describes time rather than memory/storage growth.
- **(c) 0/3 —** Does not define tractability, intractability or resource limits as input grows.
- **(d) 0/2 —** Does not explain how different algorithms can have different growth while returning the same result.

Selected task C5

Eight-bit two's-complement interpretation, representation of −45 and calculation of 54 - 19.

Example X

(a) 00 101 101 has signed value +45 because its most significant bit is 0, indicating a non-negative value.

(b) Invert 00 101 101 to get 11 010 010, then add 1 to get 11 010 011. Therefore −45 is 11 010 011 in eight-bit two's complement.

(c) 19 is 00 010 011; invert to 11 101 100 and add 1 to get 11 101 101. 54 is 00 110 110. I get 00 110 110 + 11 101 101 = 1 00 100 010; discarding the carry gives 00 100 010, or 34.

Reference judgement: 7/10. Proposed subpart judgements follow.

- **(a) 2/2 —** Credit: Gives signed value +45; Explains MSB 0 indicates a non-negative value.
- **(b) 3/3 —** Credit: Inverts 00 101 101 to 11 010 010; Adds one to obtain 11 010 011; Identifies 11 010 011 as −45.
- **(c) 2/5 —** Credit: Represents 19 as 00 010 011 and inverts to 11 101 100; Adds one to obtain 11 101 101. Not credited: Shows correct addition 00 110 110 + 11 101 101 = 1 00 100 011; Discards carry to obtain 00 100 011; Gives decimal result 35.

Example Y

(a) The signed value is +45. The most significant bit is the sign bit.

(b) Invert 00 101 101 to get 11 010 010; that is the representation of −45.

(c) 19 is 00 010 011; its inversion is 11 101 100. I did not complete the addition.

Reference judgement: 3/10. The candidate reads the positive signed value and can begin two's-complement conversion, but omits the add-one step and does not complete the subtraction calculation.

- **(a) 1/2 —** The signed value is correct; the significance of an MSB of zero is not explained.
- **(b) 1/3 —** The inversion is correct, but one must add one to obtain 11 010 011 before identifying −45.
- **(c) 1/5 —** The representation and inversion are correct; the add-one, binary sum, carry handling and decimal result are absent.

Example Z

(a) 00 101 101 is +45. The most significant bit is 0, so it is positive.

(b) Invert 00 101 101 to get 11 010 010. I am unsure what to do after that.

(c) [No response]

Reference judgement: 3/10. Proposed subpart judgements follow.

- **(a) 2/2 —** Correct signed value and interpretation of the most significant bit.
- **(b) 1/3 —** Correct inversion step; does not add one or identify the final representation as −45.
- **(c) 0/5 —** No two's-complement working or result for 54 - 19 is provided.