2026 A/L ICT
Model Marking Scheme

English Medium · Paper I and Paper II

Prepared for the supplied 2026 paper, using the available 2024 and 2025 marking schemes as references. Version 1.0 · 9 November 2026.

Status and intended use

Independent model scheme. Not an official Department of Examinations marking key. The question-level totals follow the supplied paper. Detailed component allocations and proposed treatments of defective questions are editorial recommendations.

Includes all 50 multiple-choice questions and all ten Paper II questions. Paper II Part A comprises Questions 1–4; Part B comprises Questions 5–10, of which candidates answer four.

Section	Maximum / use
Paper I	50 raw marks: one per MCQ, subject to the defective-item policy
Paper II Part A	40 marks: all four questions, 10 each
Paper II Part B	60 marks: four questions, 15 each
Paper II total	100 marks; all six optional questions are answered in this scheme

Read these qualifications first

The supplied 2024 reference has **10 blank PDF pages** and cannot support a claim of complete year-to-year comparison. The 2025 reference is more usable, but its page numbering skips the embedded examination-paper section.

The 2026 Paper II contains **14 actual pages**, despite a cover statement of 16. All Questions 1–10 and their subparts are present in the supplied file; there is no basis for inventing two additional pages.

Important uncertainties are identified for MCQs 17, 29, 30 and 38; Paper II 5(b), 7(b)(ii–iii), 8(a) and 9. Resolve these consistently before using the scheme for formal assessment.

How to use the scheme

Use the MCQ key with its explanations, not the option numbers alone. For Paper II, award marks for the stated evidence and technically equivalent alternatives. Apply the same error-handling policy to every candidate. Preserve fractional marks until the final Paper II total.

Notation: NOT(A) or A′ means complement; juxtaposition or AND means conjunction; + or OR means disjunction. Primary keys in relational schemas are underlined.

Reference analysis: 2024 and 2025

Feature	2024 reference	2025 reference / implication
Evidence quality	31 PDF pages; 10 are blank. Some diagrams are poor scans.	40 PDF pages with a usable text layer and detailed question-specific marking notes.
Partial credit	Uses 0.5-mark components, essential keywords, alternative answers and dependencies.	Frequently separates answer, justification, syntax and completeness into components.
Programming	Accepts algorithmic alternatives and specifies exact case/spelling in selected blanks.	Explicitly permits equivalent functionality; requires correct indentation and syntax for the component earned.
Diagrams	Separately credits entities, cardinalities, relationship attributes and keys.	Credits entity completeness, cardinalities, descriptive attributes and underlined primary keys.
Networking	Accepts valid alternative subnet assignments; often marks whole rows.	Uses full-row allocation for subnet tables and accepts alternative valid layouts.
Defective questions	Contains an explicit concession for a misprinted question (PDF p22, printed p49).	Detailed exceptions are question-specific; they are not universal rules.
Totals	Defers rounding of half marks until final Paper II total.	Same approach; mark all attempted answers, then exclude lower-scoring excess answers.

What is carried forward

This model uses short marking points, labelled alternatives, component totals and explicit code requirements. It does not copy old option keys, carry forward obsolete variable names, or assume an old question's "no partial marks" rule applies to a new question.

For code, the relevant component must be meaningful and syntactically sound. A separate correct component may still earn credit. For explanations, equivalent wording is acceptable unless the question explicitly requires exact wording, an option label or a particular identifier.

Limits of the comparison

Blank pages in the supplied 2024 PDF: **3, 5, 7, 9, 11, 13, 15, 17, 29 and 31**. The visible 2024 marking section runs through printed pages 31–57 with gaps. Its MCQ answer key is not recoverable from this supplied copy. No missing answers have been inferred from blank pages.

The 2025 reference's physical PDF pages 6–39 correspond to printed pages 30–63. Specific precedents used include PDF pp7–9 (SQL), 13–18 (programming/OS), 19–25 (logic/networking), 30–36 (Python/database design), and 37–39 (web applications).

Marking rules and score control

Recommended model rules

1. Award each component once. Do not exceed a subpart’s printed maximum. Accept a correct equivalent method unless the question restricts the method, such as sequential search, a while loop, or NAND-only gates.

2. For code, preserve Python indentation and keyword case. SQL keyword case is immaterial. Variable names may differ when not fixed by the question, but must be consistent. Exact blank labels and supplied attribute names should be retained.

3. Use error-carried-forward credit where a later calculation correctly follows an earlier numerical answer and the same error would otherwise be penalised twice. Do not carry forward an error that eliminates the skill being tested.

4. If extra Part B questions are answered, mark all and count the best four, following the reference schemes. Record raw fractional marks and round only the final Paper II total upward to a whole number if adopting their reporting convention.

5. Do not silently guess corrections. For a defective MCQ already administered, a uniform neutralisation policy is preferable to requiring candidates to infer the typo. Suggested policies here are model recommendations, not official concessions.

Question	Component totals	Total
1	5 + 2 + 3	10
2	8 + 2	10
3	3 + 3 + 4	10
4	5 + 2 + 2 + 1	10
5	3 + 6 + 6	15
6	2 + 2 + 5 + 2 + 4	15
7	8 + 4 + 3	15*
8	8 + 7	15
9	5 + 4 + 4 + 2	15
10	5 + 7 + 3	15

*Q7(b)(ii) has no printed allocation. A model allocation of **1 mark** makes Q7 total 15. This is an inference and must be confirmed by the setter before formal use.

Paper I scoring

One raw mark per valid response, no negative marking. Blank or multiple selections normally score zero. For MCQs 30 and 38, the literal printed question has no listed correct answer; recommended post-administration treatment is 1 mark to every candidate for each, keeping the denominator 50. If corrected before administration, use the intended answers 1 and 1 respectively.

This document leaves Paper I and Paper II totals separate. It does not assert an official combined weighting or grade boundary.

Corrections and moderation decisions

Item	Issue and recommended treatment
MCQ 17	Option 5 is the conventional intended RSA-like answer, but “private-key encryption” is not a general definition of all asymmetric cryptography. Explain signatures separately; do not teach this as a universal rule.
MCQ 29	All INSERT statements are malformed SQL. Intended constraint-count answer: 5. Treat the listed tuples as proposed records after an explicit syntax correction; if already used, accept a sound syntax objection.
MCQ 30	The query contains DO1 (letter O), while records use D01 (zero). Literally: no rows, with no matching option. After correcting to D01: option 1.
MCQ 38	The last line reads new list, not new_list. Literally SyntaxError; no error option exists. With the underscore restored: option 1.
Q5(b)	The long overbar on the final term is ambiguous and visually reaches the c. The valid identity is Xc′ + X′c, with X = a′b + ab′. If the bar includes Xc, the printed identity is false. See the proof and counterexample.
Q7(b)(ii)	No printed mark allocation. Proposed 1 mark.
Q7(b)(iii)	Agent autonomy cannot be ranked from job titles alone. ITA is a likely intended answer, but the scenario lacks design constraints establishing autonomy. Accept a reasoned alternative or neutralise the item.
Q8(a)	Instruction says use only bracketed terms for entities/relationships, but brackets supply attribute names only. Accept sensible entity/relationship names; keep the supplied attribute names.
Q9	Worked example gives 12 days → Rs.225. This requires the literal over-10 rule 50 + 25(d−5), not the usual progressive 50 + 25(d−10). Main key follows the printed example; an alternative policy is documented.
Paper II cover	Says 16 pages, but supplied PDF contains 14. All numbered questions are present. Do not infer additional questions or marks.

Other wording notes

MCQ 18 assumes Internet access using private IPv4 addresses; a purely internal LAN does not inherently need NAT. MCQ 21 uses the usual closed-system interpretation for a sealed flask; thermal exchange is not excluded. Q3 processes **non-negative** inputs, including zero, despite its “positive numbers” message. MCQ 43 asks for a class selector; .notice can match other elements of the same class if they exist.

Q9 records describe late returns, not necessarily unique members. The supplied list has one entry per member, so the requested output is per record; do not add aggregation requirements not specified by the question.

Paper I: key and reasoning (1/3)

Q	Option	Reason / qualification
1	3	Fingerprint verification requires an immediate response.
2	4	Presence → type → range.
3	1	CD-ROM is read-only.
4	1	A uses a control signal; B uses the data bus and C the address bus.
5	4	The program counter holds the address of the next instruction.
6	1	−1 in 8-bit two’s complement is 11111111.
7	2	4E 41 4D 45 gives N A M E.
8	2	A and B are true. Octal 645 = binary 110100101, not 110100111.
9	4	14 = 1110; 0.3125 = 1/4 + 1/16 = .0101.
10	5	1 XOR B = NOT(B).
11	1	AB + AB′ + AC = A(B+B′) + AC = A + AC = A.
12	1	UTP is the expected unshielded-copper answer; fibre is immune to electromagnetic pickup. The broad word “media” is imprecise because radio signals can also suffer interference.
13	1	Two subnet bits divide the entire IPv4 /0 space into four blocks: /2.
14	4	An Ethernet switch forwards frames.
15	2	TCP supplies reliable, ordered delivery; encryption is not inherent to TCP.
16	4	D → A → E → C → B.
17	5*	Intended conventional private/public key-pair description. See the limitation on the corrections page.

*Asterisk denotes a qualification or a defective printed item. Apply the corrections and moderation policy before scoring.

Paper I: key and reasoning (2/3)

Q	Option	Reason / qualification
18	3*	Private IPv4 addresses normally require translation to a public address for IPv4 Internet access. Merely having servers or wireless links does not require NAT.
19	5	ES → stored expert knowledge; MIS → routine summaries; CMS → website content editing.
20	4	Prototyping clarifies uncertain requirements through sample interfaces and feedback.
21	3	A and B: sealed flask as closed; school as open/man-made. Natural systems need not be living.
22	1	A operational; B economic; C organizational.
23	3	Selecting seats is a system function; other options describe quality/capacity.
24	1	White-box; integration; acceptance.
25	3	A and B: parallel operation reduces transition risk; COTS may have recurring fees and may need adaptation.
26	5	Network; hierarchical; relational.
27	2	A table can have multiple UNIQUE constraints but one PRIMARY KEY, possibly composite.
28	5	CourseID is unique and non-null. CourseName is UNIQUE but nullable as declared (MySQL interpretation).
29	5*	Proposed row violates the primary-key non-null rule and NIC uniqueness, but not Age or Name. Printed SQL syntax is invalid.
30	1* / invalid	Intended D01 query returns Kasun, 85000. Literal DO1 returns no rows. No listed literal answer.
31	2	M:N enrollment with EnrollmentDate attached to the relationship.
32	2	OrderID alone is part of the composite key and determines non-key CustomerID: partial dependency. Options 1 and 3 are duplicated.
33	1	0NF D; 1NF B; 2NF C; 3NF A. These are simplified distinguishing features, not full formal definitions.
34	2	Only B is correct: the linker combines object files/libraries. Assembler and loader descriptions are wrong.

*Asterisk denotes a qualification or a defective printed item. Apply the corrections and moderation policy before scoring.

Paper I: key and reasoning (3/3)

Q	Option	Reason / qualification
35	4	True: 3**2=9; 2*9//5=3; 5+3<=8.
36	3	4 5. Iteration 1 adds 2, then x becomes 3; iteration 2 adds 3, then x becomes 4.
37	3	[15, 60, 15]. update sets index 1 to 50; restock adds 10 to indices 0 and 1.
38	1* / invalid	After correcting new list to new_list: ['Saman', 70] [100]. As printed the program cannot parse.
39	4	A string → 4; B list → 1; C tuple → 3; D dictionary → 2.
40	5	Prints 5, 3, 1. The final decrement occurs after printing.
41	4	Q and S handle purchases or private authenticated results.
42	1	A subdirectory; B parent directory; C external absolute URL; D same-page fragment.
43	2	.notice selects the supplied class. p.notice would restrict the class selector to paragraphs, but is not listed.
44	5	All three: ID colour rules override the paragraph/class colours; .text still supplies bold.
45	1	A description list shows HTML and an indented description without bullets or numbering.
46	5	Host, username, password, database. The fifth argument is port; the options ask about the first four.
47	2	LDR → analog input; LED → digital output; reed switch → digital input.
48	1	A piezo buzzer is an actuator. An LDR is not a diode; the integrated CPU/memory/I/O description is of a microcontroller.
49	3	A qubit can be in superposition; measurement gives a classical outcome.
50	3	Agents act autonomously on local perceptions and communicate when needed.

*Asterisk denotes a qualification or a defective printed item. Apply the corrections and moderation policy before scoring.

Paper II · Q1: Web development [10]

1(a) HTML/CSS [5]

Subpart	Answer	Marks
(i)	1 E (action); 2 H (method); 3 C (label); 4 A (text); 5 F (select); 6 I (option); 7 O (date); 8 G (submit).	8 × 0.5 = 4
(ii)	Missing colon after background-color. Correct: `background-color: blue;`	0.5
(iii)	Browser ignores the invalid colour declaration: the submit control does not acquire the specified blue background from that rule. Other valid declarations, including padding, still apply.	0.5

The question requests option labels for (i). For formative use, an unambiguous correct keyword can also be accepted. The repeated label 3 and repeated label 4 each earn credit once, not once per occurrence.

1(b) PHP blanks [2]

```
1: bookid
2: date
3: $var1
4: $var2
```

0.5 each, in order. The first two are the form control names inside the existing quotes; the last two are PHP variables. The name field, not the id field, determines the POST key.

1(c)(i) Expected rendered table [2]

Library Book Records

Book ID	Category	Title	Status
B101	Science	Mechanics	Available
B102		Chemistry Basics	Borrowed
B103	ICT	Web Design	Available
B104		Programming Basics	Borrowed

Model allocation: correct heading/column headers 0.5; all four data rows 0.5; Science spans two rows 0.5; ICT spans two rows 0.5. Exact font or pixel spacing is not required.

1(c)(ii) Add the fifth book [1]

```
<tr><td>B105</td><td>CSS Basics</td><td>Available</td></tr>
```

Insert after B104 and before the closing table tag (0.5). Change the ICT cell in the B103 row from `rowspan="2"` to `rowspan="3"` (0.5). Do not add a new category cell to the B105 row.

Paper II · Q2: Systems and e-business [10]

2(a) Library registration [8]

Part	Expected evidence	Marks
(i)	LibMIS: student. SchoolLIB: librarian.	0.5 + 0.5
(ii)	The librarian manually locates/checks paper student records before entering each registration; serial manual verification creates the bottleneck.	1
(iii)	LibMIS validates automatically against the central school database; SchoolLIB verifies manually against paper records. Credit both sides of the comparison.	1
(iv)	Replace librarian-only entry with student self-registration using the online registration page. This must be distinct from the verification change.	1
(v)	Student submits details online → automatic central-database validation → account creation if valid → email notification of the result. Self-service and automated validation are both required.	1
(vi)	P = Registration details; Q = Student details; R = User account details; S = Register user.	1
(vii)	Black-box testing (1): validates external input/output behaviour without inspecting internal code (1). Valid/invalid format cases may use equivalence partitioning, but the broad technique is black-box.	2

For (vi), proposed allocation: all four correct = 1; two or three correct = 0.5; fewer than two = 0. This follows the style of the 2025 four-label item and avoids unannounced quarter-mark rounding.

For (v), award 0.5 for self-service initiation and 0.5 for the automatic verification/account workflow. Notification is part of the complete model answer; do not insist on an exact sentence.

2(b) Fill the blanks [2]

Part	Answer / alternatives	Marks
(i)	Third-party payment service / third-party payment processor / digital wallet (such as PayPal). “Secure electronic payment service” is acceptable if clearly identified as an intermediary; PayPal alone merely repeats the example.	1
(ii)	G2C: government-to-citizen electronic service. Accept C2G if explicitly framed as the citizen’s application/payment to government; the question does not specify transaction direction.	1

Do not classify (b)(ii) as B2C merely because a fee is paid. It is a government–citizen interaction.

Paper II · Q3: Python and algorithms [10]

3(a) Maximum of entered values [3]

(i) 25 [1]. The negative sentinel ends input and is not included.
(ii) Finds and prints the largest non-negative integer entered before the first negative integer [1].
(iii) found records whether at least one eligible, non-negative input was processed, so the program can distinguish an empty eligible sequence from a valid maximum of zero [1].

Accuracy note: zero is included by n >= 0. The printed “No positive numbers” wording is misleading; “No non-negative numbers” would describe the implemented test.

3(b) Average program blanks [3]

```
1: 0
2: total + n
3: count + 1
4: n = int(input())
5: count
6: total / count
```

0.5 per blank. count guards against division by zero. The sentinel is excluded, but zero-valued inputs are included. Accept equivalent expressions preserving the surrounding assignment syntax.

3(c)(i) Flowchart blanks [2]

```
1: reverse
2: n > 0
3: n % 10
4: reverse * 10 + digit
5: n = n // 10
```

Award 0.5 for correct initialization and loop condition together; 0.5 for digit extraction; 0.5 for accumulation; 0.5 for integer-division update. DIV 10 and MOD 10 are valid flowchart notation.

3(c)(ii) Python implementation [2]

```
n = int(input())
reverse = 0
while n > 0:
    digit = n % 10
    reverse = reverse * 10 + digit
    n = n // 10
print(reverse)
```

0.5 input and initialization; 0.5 correctly controlled while loop and terminating integer update; 0.5 digit extraction and accumulation; 0.5 output after the loop. Accept an equivalent arithmetic while-loop implementation. String reversal alone does not implement the specified flowchart.

Checks: 1234 → 4321; 120 → 21; 1 → 1. Leading zeroes are not retained in an integer result. No negative-input handling is required because the stated input is positive.

Paper II · Q4: Operating systems [10]

4(a) File allocation [5]

Part	Answer	Marks
(i)	File 1: 20 → 22 → −1. File 2: 21 → 24 → 25 → −1. File 3: 23 → 28 → 26 → 27 → −1.	0.5; 0.5; 1
(ii)	File 1 has 2 blocks; File 3 has 4. Total = (2 + 4) × 4 KB = **24 KB** (24 KiB if KB denotes 1024 bytes).	1
(iii)	**Indexed allocation** (1): an index gives direct access to a desired block and supports non-contiguous growth, whereas a linked chain normally requires traversal for random access (1).	2

The −1 entry is an end marker, not an allocated block. File 3 starts at 23 because it is the remaining chain head in the complete three-file fragment. For (ii), 0.5 for the six-block calculation and 0.5 for 24 KB.

4(b) Context switch P3 → P2 [2]

(i) Save P3's CPU execution context—program counter, register values, stack pointer/status as applicable—in **P3's PCB** [1: data 0.5, correct PCB 0.5]. Restoring P2's saved state completes the switch, but does not replace the requested explanation of what is saved.

(ii) Saving/restoring state and changing the running process consumes CPU time without advancing the user processes' useful work [1].

4(c) Address translation [2]

Part	Calculation and answer	Marks
(i)	48-bit byte address → maximum virtual address space **2^48 bytes**.	0.5
(ii)	Physical frames = 2^33 / 2^12 = 2^21 → **21 frame-number bits**. Additional status/control bits are separate.	0.5
(iii)	8193 = 2 × 4096 + 1, so offset = 1. Physical address = frame 2 × 4096 + 1 = **8193**.	1

In (iii), the equality of virtual and physical numerical values is coincidental: virtual page 2 happens to map to physical frame 2. Award 0.5 for the offset and 0.5 for the resulting address.

4(d) Printer driver [1]

A device driver translates the OS's generic I/O requests into the printer-specific commands/protocol and provides the interface needed to communicate with that hardware. A suitable answer must connect the driver to hardware-specific communication; "it installs the printer" alone is insufficient.

Paper II · Q5(a–b): Boolean algebra [9]

5(a) De Morgan’s theorems [3]

```
NOT(A AND B) = NOT(A) OR NOT(B)
NOT(A OR B)  = NOT(A) AND NOT(B)
```

A	B	NOT(AB)	A′ + B′	NOT(A+B)	A′B′
0	0	1	1	1	1
0	1	1	1	0	0
1	0	1	1	0	0
1	1	0	0	0	0

0.5 for each correct theorem statement; 1 for complete truth-table proof of the first; 1 for the second. Accept valid algebraic proofs instead. The matching columns prove the identities for every input combination.

5(b) Factor and prove the parity identity [6]

Let **X = a′b + ab′**. The correct identity for the printed left-hand side is **Xc′ + X′c**. Use prime notation here to make the scope of each complement unambiguous.

```
L = a'b'c + a'bc' + ab'c' + abc
  = (a'b + ab')c' + (a'b' + ab)c

X' = (a'b + ab')'
   = (a + b')(a' + b)
   = aa' + ab + a'b' + b'b
   = ab + a'b'

Therefore L = Xc' + X'c
            = (a'b + ab')c' + (a'b + ab')'c
```

Model allocation: valid grouping of the four minterms [2]; correct De Morgan transformation and distribution proving X′ = ab + a′b′ [2]; substitution and correct final identity [2]. Award 1 within each component for a correct substantial intermediate step. A truth table alone does not satisfy the requested Boolean-algebra method.

Printed-overbar qualification

In the supplied page the long bar on the final term visually reaches the final c. If it means **NOT(Xc)**, the equation is false: at a=b=c=0 the left side is 0 but Xc′ + NOT(Xc) is 1. No valid proof of that literal identity exists.

Before assessment, print the RHS as **(a′b + ab′)c′ + (a′b + ab′)′c**. If already administered, accept the correct parity derivation; a correct counterexample to the literal overbar interpretation should not be penalised. The setter must approve a uniform concession policy.

Paper II · Q5(c): NAND access circuit [6]

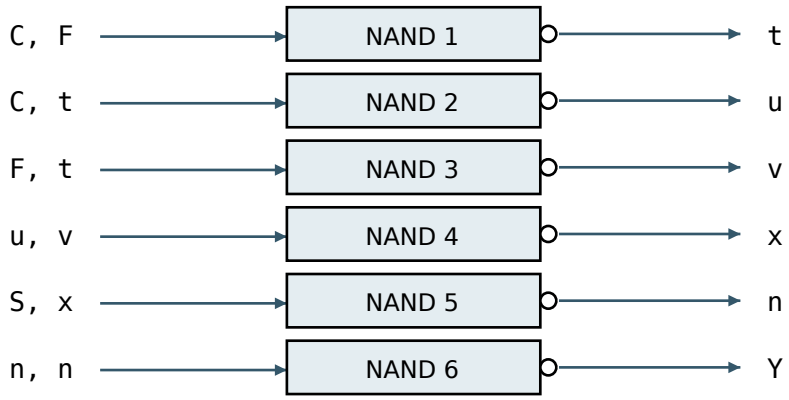
Assume active-high inputs: **S** = supervisor key inserted; **C** = clerk key inserted; **F** = supervisor fingerprint authenticated. Output **Y=1** opens the door. “Fingerprint” means a valid supervisor match, not merely contact with the scanner.

S	C	F	Y
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Y = SC′F + SCF′ = S(C XOR F). Both permitted combinations require S. The all-three-input combination must produce zero, so SC + SF is wrong.

NAND-only implementation: six two-input gates

The following is a labelled circuit/net diagram. Identically named signals are connected; each box is a NAND gate. The final gate has its two inputs tied together.



t = (CF)′; u = (Ct)′; v = (Ft)′; x = (uv)′ = C XOR F; n = (Sx)′; Y = (nn)′ = Sx. The six-gate construction was checked against all eight truth-table rows.

Allocation: defined inputs/active levels [1]; correct truth table [2, 0.25 per row]; correct expression [1]; connected NAND-only implementation [2: XOR stage 1, supervisor enable/inversion 1]. Retain quarter marks until the final total if this row-based allocation is used. An equivalent NAND design, including a correct three-input-NAND sum-of-products realization, is acceptable.

Paper II · Q6: Networking [15]

6(a–b) NAT and address-space size [4]

(a) [2] NAT translates private internal IPv4 addresses to public addresses for communication with the public Internet. Credit translation [1] and Internet access/public-address conservation [1]. A single public IP shared using port translation is an acceptable example. NAT is not encryption and is not a substitute for firewall policy.

(b) [2] 32 − 16 = 16 host bits; 2^16 = **65,536 total addresses**. Credit exponent/method [1], total [1]. Do not subtract two when asked for total addresses.

6(c) Subnet plan [5]

The unallocated remainder is 192.168.192.0/18, spanning third octets 192–255. Halving it gives two /19 networks, each with 8192 addresses and 8190 conventional usable host addresses.

Department	Network / mask	Usable range	Broadcast
Production	192.168.0.0/17 255.255.128.0	192.168.0.1 – 192.168.127.254	192.168.127.255
Marketing	192.168.128.0/19 255.255.224.0	192.168.128.1 – 192.168.159.254	192.168.159.255
HRM	192.168.160.0/19 255.255.224.0	192.168.160.1 – 192.168.191.254	192.168.191.255
Accounting	192.168.192.0/19 255.255.224.0	192.168.192.1 – 192.168.223.254	192.168.223.255
IT	192.168.224.0/19 255.255.224.0	192.168.224.1 – 192.168.255.254	192.168.255.255

1 per fully correct row, consistent with the reference subnet-table style. Accounting and IT may exchange their two /19 assignments. The original three allocations must remain unchanged; no overlaps or gaps in the allocated /16 are allowed.

6(d) Communication media [2]

Fibre-optic inter-building links are a strong answer: high bandwidth, longer reach, resistance to electromagnetic interference and electrical isolation. Award 1 for an appropriate medium and 1 for a relevant suitability reason. Also accept a properly justified point-to-point wireless/microwave link with line of sight, or two suitable named media at 1 each. UTP is conditional on distance and installation limits; building distances are not supplied.

6(e) Firewall placement [4]

```
Internet / ISP router ↔ FIREWALL ↔ core LAN / department subnets
```

Place at the boundary between the external Internet connection and the protected internal LAN; it may be integrated with the edge router. Allocation: correct perimeter location [2]; filters traffic according to security rules [1]; protects all departments because external traffic must traverse it [1]. A firewall on one departmental endpoint alone does not protect the entire organisation.

Paper II · Q7(a): Embedded system [8]

7(a)(i) Complete the Arduino code [4]

```
A: 2
B: INPUT
C: HIGH
D: HIGH
```

1 per answer. The PIR OUT wire connects to digital **pin 2**, confirmed from the circuit image. The LED uses pin 8. Use the symbolic INPUT and HIGH constants; correct equivalent numeric constants may receive credit if local marking policy permits them.

7(a)(ii) Add the piezo buzzer [2]

```
void loop() {
    int motion = digitalRead(pirPin);
    if (motion == HIGH) {
        digitalWrite(ledPin, HIGH);
        tone(9, 1000);
        delay(10000);
        noTone(9);
        digitalWrite(ledPin, LOW);
    }
}
```

Start an audible tone on pin 9 when motion is detected [1]; stop the tone after the same ten-second interval as the LED [1]. Any suitable audible frequency is acceptable. `tone(9, 1000, 10000);` with matching LED timing is also valid.

A passive piezo needs an oscillating signal. Merely writing HIGH does not generate a sustained tone. Accept digitalWrite(HIGH/LOW) only if the candidate explicitly assumes an active, internally oscillating buzzer. Do not demand modifications to setup when a correct tone/noTone implementation answers the requested loop change.

7(a)(iii) Night-only detection [2]

LDR / light sensor [1], connected through a suitable voltage-divider circuit to an analog input such as **A0** [1]. A module with an analog output can connect directly as specified by its interface. A fixed resistor is normally needed with a bare LDR.

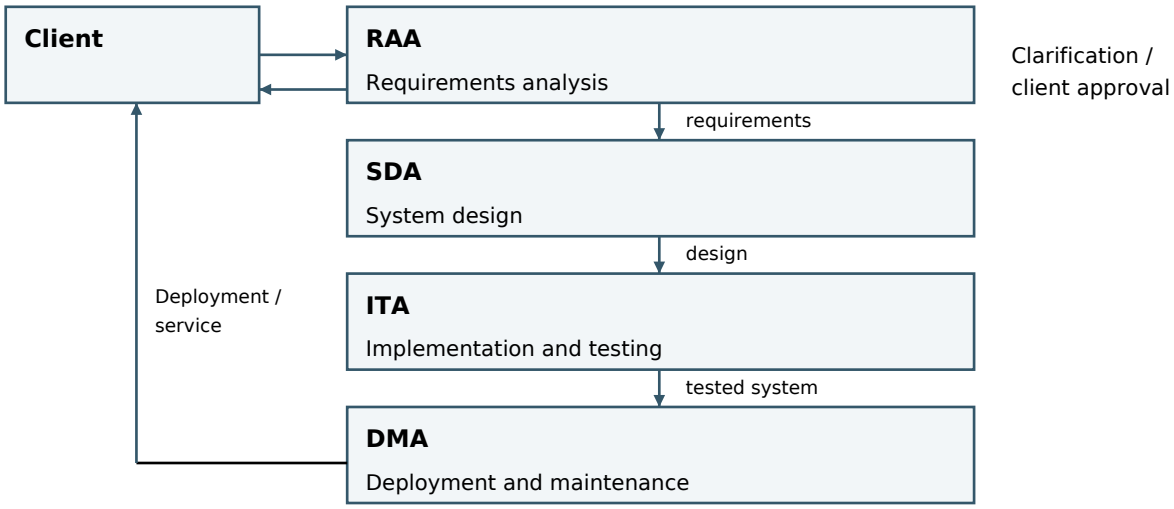
The operating condition would become motion detected AND dark enough. The comparison direction depends on which side of the voltage divider contains the LDR, so do not insist on an arbitrary threshold direction when the wiring is unspecified.

Technical reference

Arduino’s tone() documentation describes a square-wave output with an optional duration and noTone() to stop it. Source: Arduino language reference: tone(). This was checked to avoid treating a passive piezo as a DC-driven sounder.

Paper II · Q7(b–c): Agents and quantum [7]

7(b)(i) Multi-agent waterfall workflow [2]



Award 1 for all four agents in the correct waterfall order; 1 for client interaction and labelled handoffs (requirements → design → implementation/test → deployment). Feedback for clarification is acceptable; do not turn the principal workflow into an unrelated parallel pipeline.

7(b)(ii) Interface agent [1 proposed]

RAA, because it obtains/clarifies the client's requirements. Accept another agent only if the candidate's diagram explicitly gives it a credible client-facing role. The question omits the printed marks; 1 is inferred to complete the 15-mark question.

7(b)(iii) Lower autonomy [1]

Likely intended: ITA, assuming it must implement and test the design supplied by SDA. However, dependency on an input is not evidence of less self-autonomy; the scenario gives no constraints on independent decisions. Accept ITA under that assumption, a reasoned SDA answer under a stated implementation, or "cannot be determined from the information supplied." Do not present an unconditional ranking as established fact.

7(c)(i) Bit and qubit [1]

A classical bit is in one of two states, 0 or 1. A qubit can occupy a superposition $\alpha|0\rangle + \beta|1\rangle$, with $|\alpha|^2 + |\beta|^2 = 1$; measurement in this basis gives 0 or 1. Award 0.5 for the classical distinction and 0.5 for superposition. Formal amplitude notation is not required.

7(c)(ii) Public-key security challenge [2]

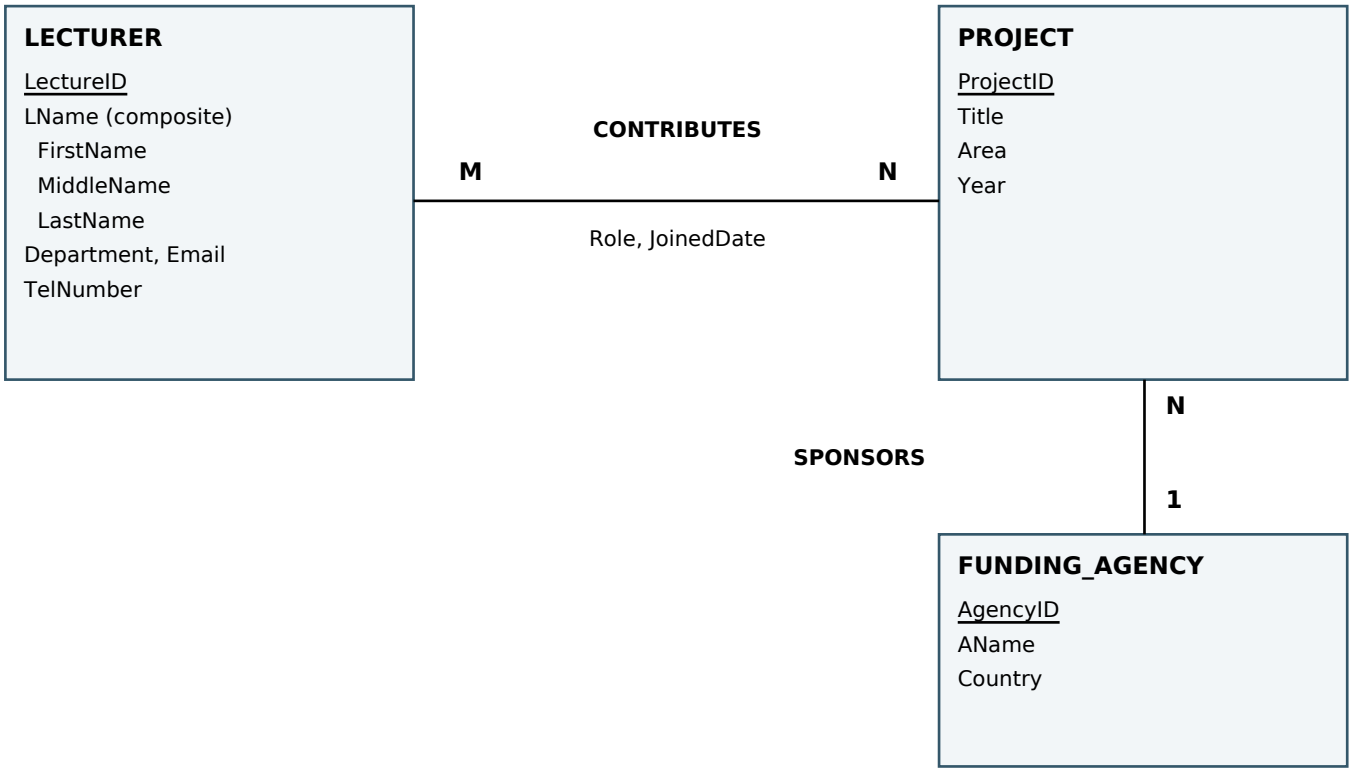
Sufficiently capable, fault-tolerant quantum computers could efficiently solve certain problems underlying RSA and discrete-logarithm cryptosystems, using algorithms such as Shor's [1]. Recovering secret keys would compromise confidentiality and/or signatures in those systems [1]. Superposition alone is not an explanation that all problems are instantly solved, and not every encryption scheme is broken.

Source: NIST: What Is Post-Quantum Cryptography?. Algorithm names are enrichment, not mandatory for full syllabus-level credit.

Paper II · Q8(a): ER design and schema [8]

8(a)(i) Entity–relationship model [4]

The diagram below uses entity boxes listing their attributes and labelled relationship links. Underlined identifiers are keys. LName is composite; Role and JoinedDate belong to the lecturer–project relationship, not to either entity in isolation.



Each project has exactly one funding agency; an agency may sponsor many projects. Lecturer/project participation is M:N; optional participation is consistent with “may.” If using Chen notation, show LName with three component ovals and attach Role/JoinedDate to the relationship diamond.

Allocation: three correct entities with attributes/keys, 0.5 each [1.5]; composite LName [0.5]; M:N with both descriptive attributes [1]; agency 1:N, project exactly one agency [1]. Entity/relationship names were not supplied in brackets, so reasonable names are necessary and acceptable.

8(a)(ii) Relational schema [4]

LECTURER(LectureID, FirstName, MiddleName, LastName, Department, Email, TelNumber)
PROJECT(ProjectID, Title, Area, Year, AgencyID)
FUNDING_AGENCY(AgencyID, AName, Country)
CONTRIBUTES(LectureID, ProjectID, Role, JoinedDate)

```
PROJECT.AgencyID          → FUNDING_AGENCY.AgencyID
CONTRIBUTES.LectureID     → LECTURER.LectureID
CONTRIBUTES.ProjectID     → PROJECT.ProjectID
```

0.5 per correct table and attribute set [2]; all correct underlined keys [1]; all three correct foreign-key arrows to referenced keys [1]. Split the composite name into atomic components; storing a redundant LName as well is unnecessary. Preserve the source spelling **LectureID** rather than silently changing it to LecturerID.

Paper II · Q8(b): Normalisation [7]

8(b)(i) Two partial dependencies [2]

```
StudentID → StudentName
ClubID    → ClubName
```

1 each. ClubID → AdvisorID is also an acceptable second example. The key is (StudentID, ClubID), and a partial dependency determines a non-key attribute from a proper subset of that composite key.

8(b)(ii) Transitive dependency [1]

```
ClubID → AdvisorID → AdvisorName
```

Accept the fully expanded key-level chain: (StudentID, ClubID) → AdvisorID → AdvisorName, with AdvisorID → AdvisorName explained. AdvisorID is not a superkey of CLUB_MEMBERSHIP. Writing AdvisorID → AdvisorName is acceptable if clearly identified as the non-key-to-non-key dependency causing transitivity.

8(b)(iii) 3NF decomposition [4]

STUDENT(StudentID, StudentName)
ADVISOR(AdvisorID, AdvisorName)
CLUB(ClubID, ClubName, AdvisorID)
CLUB_MEMBERSHIP(StudentID, ClubID, JoiningDate)

1 per correct table, including its underlined key. Sensible alternative table names are acceptable. The source attribute names should be retained.

```
CLUB.AdvisorID             → ADVISOR.AdvisorID
CLUB_MEMBERSHIP.StudentID  → STUDENT.StudentID
CLUB_MEMBERSHIP.ClubID     → CLUB.ClubID
```

These foreign-key references clarify the solution; the question does not separately require arrows here, so do not deduct for their absence.

Why this is 3NF

Student details depend on StudentID; club details and its advisor identifier depend on ClubID; advisor name depends on AdvisorID. JoiningDate describes a particular student–club membership and remains with the composite key. Removing AdvisorName from CLUB removes the transitive dependency through AdvisorID.

The model assumes one membership record per student–club pair, as explicitly stated. Do not introduce a new MembershipID or multiple joining episodes: those would change the stated requirements.

Common errors to reject

Do not put JoiningDate in STUDENT or CLUB, because a student can join different clubs on different dates. Do not put AdvisorName back into CLUB. Do not omit either part of the membership primary key. Do not claim that merely splitting the table into any four tables proves 3NF.

Paper II · Q9: Fine calculation [15]

Main key follows the printed example: for more than 10 days, fine = 50 + 25 × (lateDays − 5), capped at 500. This yields the paper's 12-day example of 225. The resulting jump after 10 days is acknowledged; it is not silently "fixed."

9(a) calculateFine [5]

```
def calculateFine(lateDays):
    if lateDays <= 5:
        return 0
    elif lateDays <= 10:
        return (lateDays - 5) * 10
    else:
        fine = 5 * 10 + (lateDays - 5) * 25
        return min(fine, 500)
```

1 function definition/parameter/returned result; 1 no-fine interval; 1 6–10 calculation; 1 over-10 calculation matching the example; 1 correct maximum of 500. Equivalent if/else or cap logic is acceptable. Valid input is assumed to be a non-negative number of days.

9(b) Build fineList [4]

```
lateReturns = [("M001", 1), ("M002", 5), ("M003", 10),
               ("M004", 20), ("M005", 30)]
fineList = []
for memberNo, lateDays in lateReturns:
    fine = calculateFine(lateDays)
    if fine > 0:
        fineList.append((memberNo, fine))
```

1 list initialization; 1 traversal with correct field extraction; 1 computing the fine; 1 retaining only positive fines as tuples in the requested list.

```
[("M003", 50), ("M004", 425), ("M005", 500)]
```

9(c) Sequential search [4]

```
def searchFine(fineList, memberNo):
    for number, fine in fineList:
        if number == memberNo:
            return fine
    return -1
```

1 function/parameters; 1 sequential traversal; 1 matching identifier and returning its fine; 1 returning −1 only after the search is exhausted. An early −1 inside the loop is incorrect. Do not replace the requested sequential search with a dictionary lookup.

9(d) Outputs [2]

```
0
20
300
125
```

0.5 each. The final 125 is read from the supplied search list, not recalculated.

Q9: alternative interpretation and checks

Why the ambiguity matters

The 0–5 and 6–10 rows imply five free days, followed by five days charged at Rs.10. A conventional progressive continuation would charge only days after day 10 at Rs.25. But the printed over-10 wording and the worked 12-day example instead charge Rs.25 for lateDays − 5 days, plus Rs.50.

Late days	Main key: literal example	Progressive alternative
4	0	0
7	20	20
10	50	50
11	200	75
12	225	100
15	300	175
20	425	300
23	500	375
30	500	500

Progressive alternative, if explicitly adopted

```
def calculateFine(lateDays):
    if lateDays <= 5:
        return 0
    if lateDays <= 10:
        return (lateDays - 5) * 10
    return min(50 + (lateDays - 10) * 25, 500)
```

This alternative yields fineList = [("M003", 50), ("M004", 300), ("M005", 500)] and part (d) outputs 0, 20, 175, 125. It does **not** reproduce the supplied 12-day example.

Recommended moderation

Before administration: choose a policy and correct either the table/example or the intended calculation. After administration: the main key must follow the explicit example; for fair formative assessment, also accept a consistently applied progressive interpretation with the same component marks. Do not penalise the same choice again in parts (b) and (d). This alternative-credit policy is proposed, not official.

Test coverage

The two versions were checked at day boundaries 5, 6, 10 and 11, at the printed example 12, at 15 and 20, and at the 500 cap. Sequential search must be checked for a first match, a later match, a missing member and an empty list. The supplied part (b) contains one tuple per member, so summation across repeated member entries is not required.

Paper II · Q10(a–b): Event system [12]

10(a) Requirements and lifecycle [5]

(i) [2] Functional requirement, exact wording: “The system should check existing registrations before accepting a new registration request.” [1]
Non-functional requirement, exact wording: “The system should be accessible at any time from different locations.” [1]

Accept any other exact functional requirement from the given list, such as viewing events, providing confirmation, or showing administrator counts.

(ii)(1) [1] Waterfall makes late requirement changes expensive/disruptive because completed analysis/design and implementation may need revisiting.
(ii)(2) [2] Agile/iterative development [1], because short increments and repeated client feedback accommodate newly requested reminders and feedback features [1]. Accept a justified iterative/incremental or evolutionary-prototyping approach; merely naming a model without explaining suitability earns at most 1.

10(b) PHP and SQL [7]

(i) [1] On success, $result is a **mysqli_result object representing the result set** returned by SELECT * FROM EVENT: all matching event rows and columns, possibly zero rows. It is not a single row or the connection object. The rows are then fetched by mysqli_fetch_assoc. A failure can return false or raise an exception depending on configuration; the question assumes success.

(ii) [2] Correct projection [1] and source/syntax [1]:

```
SELECT EventID, EventName FROM EVENT;
```

(iii) [2] The hidden inputs include studentID and the selected event’s eventID in the POST to register.php [1]; they carry the necessary context without visible/retyped input [1]. Hidden does **not** mean encrypted, trustworthy, or inaccessible to the user.

(iv) [2] Within the surrounding PHP string:

```
$sql = "INSERT INTO REGISTRATION
        (StudentID, EventID, RegistrationDate)
        VALUES ('$studentID', '$eventID', CURRENT_DATE)";
```

1 correct INSERT target/column list omitting auto-generated RegistrationID; 1 corresponding values and a valid current-date expression. CURDATE() is acceptable in MySQL. Correctly constructed equivalent SQL is acceptable.

The interpolation above matches the exercise’s style. In a live application, use parameterised SQL and server-side validation; the requested two marks do not require implementing that security extension. Do not use a hidden studentID as proof of identity.

Sources: PHP mysqli_query; MySQL INSERT reference.

Paper II · Q10(c) and implementation notes

10(c) Deployment [3]

(i) [1] Direct deployment replaces the old system at once: usually lower duplicate-running cost but higher operational risk if the new system fails. Parallel deployment runs old and new systems together: higher cost/workload but lower transition risk because results can be compared and the old system remains available. Award 0.5 for the cost comparison and 0.5 for the risk comparison.

(ii) [2] Generally **do not recommend direct deployment for the bank** [1]. Failure could interrupt critical banking services or corrupt/mis-handle transactions with no concurrently operating fallback; parallel or carefully phased deployment reduces that exposure [1]. A conditional recommendation for direct deployment needs a convincing, explicit recovery/rollback and validation strategy; do not credit a bare “yes.”

What the printed sample application does not prove

The code in Q10 inserts a registration and prints a success message. It does not demonstrate duplicate-registration checking, authenticated identity, permission checks, or successful-insert verification. These are real limitations of the example but are outside the requested INSERT blank.

Do not deduct the INSERT marks merely because a candidate does not rewrite the whole application. Conversely, do not award extra marks beyond the printed maximum for adding production-security features.

Answer-scheme consistency checks

Check	Result / boundary
Coverage	Paper I Q1–50 and Paper II Q1–10, including every printed subpart, are addressed.
Paper II totals	Part A = 4 × 10 = 40; any four Part B questions = 4 × 15 = 60.
Q7 allocation	One missing printed allocation explicitly identified; model 1 mark, not represented as official.
Programming	Python traces and reference algorithms checked with representative and boundary inputs. Arduino code checked against API behaviour; not physically tested.
Numerical answers	Subnet boundaries, host ranges, block chains, address translation and Boolean truth tables independently calculated.
Defective items	Literal and intended answers separated; no silent repair of MCQ30/38 or Q9.
Source limitations	Blank 2024 pages and 2026 cover-length mismatch disclosed. No official 2026 key supplied or claimed.

The scheme is suitable for teacher review and model-answer preparation. Its accuracy is bounded by the supplied paper’s wording and legibility; the flagged setter decisions should be settled before it is used as a high-stakes scoring authority.

Sources, verification and audit trail

User-supplied references

R24: 2024-AL-ICT-Paper-English-MarkingScheme-Full (1).pdf — 31 PDF pages. Analysed through rendered-page review and OCR of readable pages. Useful precedents: PDF p4 (notes/HTML), pp8–10 (algorithm alternatives), pp16–18 (logic), pp20–22 (networking and misprint treatment), p23 (Arduino), pp27–28 (database design), p30 (OS). Ten pages are blank as listed earlier.

R25: AL-ICT-2025-Paper-Marking-Scheme-English-Medium-itguru-by-Teran-Subasinghe (2).pdf — 40 PDF pages. Analysed using extracted text and the question-specific marking guidance. PDF page numbers and printed numbers differ; mappings are given in the reference-analysis section.

P26: al-2026-english_2026-10-27_22-48-15 (1).pdf — the cleaned English Medium file in the user's Past Papers folder. 25 PDF pages: Paper I pp1–11; Paper II pp12–25 (printed pp1–14). Model answers are derived from this file, not from the 2024/2025 answer keys.

Primary technical checks

PHP: mysqli constructor / connection — MCQ46 parameter order.

PHP: mysqli_query — Q10(b)(i) result object.

MySQL: INSERT — MCQ29 syntax; Q10 insert.

MySQL: CREATE INDEX — MCQ28 nullable UNIQUE column.

Python: lexical analysis — MCQ38 invalid spaced identifier.

Arduino: tone — Q7 piezo waveform.

Arduino: noTone — Q7 stopping sound.

NIST: post-quantum cryptography — Q7 quantum security explanation.

Verification performed

All eight input combinations were checked for the valid Q5(b) parity identity and the Q5(c) six-NAND circuit. Five subnets were computed with address arithmetic. The printed and corrected MCQ30 queries were evaluated against the supplied four-row data. The printed MCQ38 line was confirmed to raise SyntaxError. Python arithmetic, mutation, reversal, fine calculations and sequential search were tested. Mark subtotals were reconciled.

The attached PDF is the complete model scheme. The companion HTML contains the same text, tables and diagrams for convenient copying and browser printing. Neither source marking scheme nor the question paper was modified.