MAH MCA CET Computer Concepts

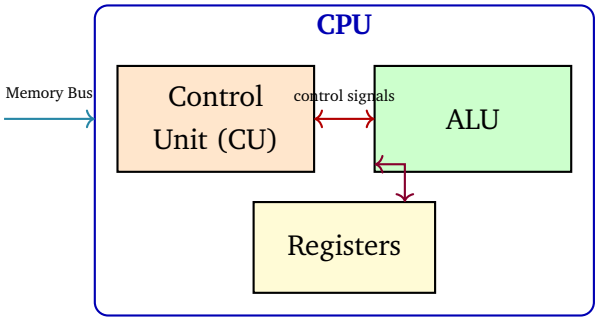
Sample Paper – 3

Duration: 23 Minutes **Maximum Marks: 50**

Instructions

- This paper contains **25** Multiple Choice Questions (Single Correct Answer), modelled on the Computer Concepts section of **MAH MCA CET**.
- Each correct answer carries **+2 marks**. There is **no negative marking** for incorrect or unattempted answers.
- Only **one** option is correct per question.
- Use of mobile phones, calculators, or electronic gadgets is strictly prohibited. Rough work may be done in the space provided in the booklet.

Q1. The diagram below shows the internal structure of a CPU. Which statement correctly describes the roles of the Control Unit (CU) and the Arithmetic Logic Unit (ALU)?



(A) The ALU permanently stores program data inside the CPU using its internal registers.

(B) The Control Unit performs all arithmetic and logical computations on data values.

(C) Registers serve exclusively for input/output (I/O) coordination between peripheral devices.

(D) The Control Unit decodes instructions and coordinates all CPU operations.

Q2. Which technology do flatbed scanners use to convert a physical document or image into digital data?

(A) A flatbed scanner uses thermal sensors to convert images into electrical signals row by row.

(B) A flatbed scanner uses CCD (Charge-Coupled Device) sensors to digitise images row by row.

(C) A flatbed scanner uses laser beams to directly convert printed text into digital pixels.

(D) A flatbed scanner captures images through ultrasonic transducer technology.

Q3. Which of the following correctly compares SRAM (Static RAM) and DRAM (Dynamic RAM)?

(A) DRAM is faster than SRAM and is therefore preferred for CPU cache memory.

(B) SRAM requires periodic refresh cycles to retain the data stored in its memory cells.

(C) SRAM is faster than DRAM, retains data without refresh, and is used in cache memory.

(D) Both SRAM and DRAM require continuous refresh cycles to prevent data loss.

Q4. Which of the following best describes the BIOS (Basic Input/Output System)?

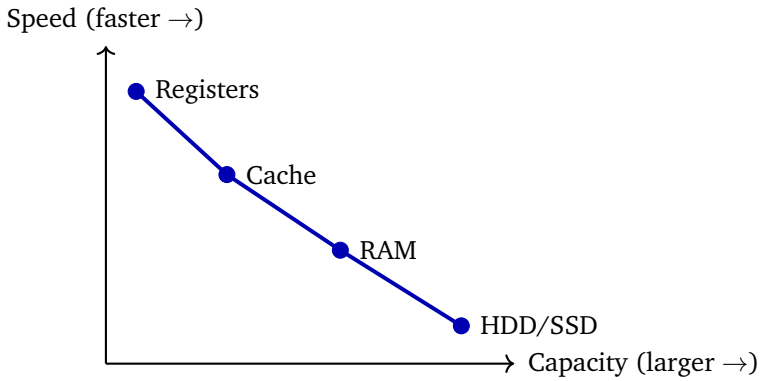
(A) BIOS is firmware stored in ROM that initialises hardware components during the boot process.

(B) BIOS is application software stored in RAM that loads the operating system kernel on demand.

(C) BIOS is a component of the operating system stored permanently on the hard disk drive.

(D) BIOS is a type of cache memory used to accelerate processor instructions during normal operation.

Q5. The chart below shows the memory hierarchy of a computer system, with access speed on the vertical axis and storage capacity on the horizontal axis. Which statement correctly describes this hierarchy?



(A) RAM is the fastest and largest memory in the hierarchy, followed by registers.

(B) Cache memory is slower than RAM but offers a significantly larger storage capacity.

(C) Secondary storage (HDD/SSD) is faster than registers but has a smaller storage capacity.

(D) Registers are the fastest but smallest memory; secondary storage is largest but slowest.

Q6. Which of the following is one of the four Coffman conditions that must hold simultaneously for a deadlock to occur in an operating system?

(A) Deadlock can occur with a single process if it requests a resource that it already holds exclusively.

(B) Circular Wait: each process in a set is waiting for a resource held by the next process in the set.

(C) Deadlocks are always resolved automatically by the operating system's scheduler without any intervention.

(D) Mutual exclusion is not required for a deadlock; any two competing processes can create a deadlock.

Q7. Evaluate the binary subtraction shown below and identify the correct result in decimal.

$$\begin{array}{r} 1101_2 \\ -\ 0101_2 \\ \hline \mathbf{1000_2} \end{array}$$

(A) 0111_2 (decimal 7)

(B) 1010_2 (decimal 10)

(C) 1000_2 (decimal 8)

(D) 0110_2 (decimal 6)

Q8. Which logic gate is considered a *universal gate*, and what property makes it so? The diagram illustrates how a NOT gate is constructed using only a NAND gate.

$\text{NAND}(A, A) = \overline{A \cdot A} = \overline{A}$ (NOT gate)

(A) NAND is a universal gate — any Boolean function can be implemented using only NAND gates.

(B) AND is a universal gate because it can implement all basic Boolean logic operations on its own.

(C) OR is a universal gate since OR gates alone can replicate both NOT and AND operations.

(D) XOR is a universal gate because it independently implements all nonlinear Boolean functions.

Q9. A stack is implemented using an array with `MAX_SIZE = 5`. The stack uses 0-based indexing, so valid positions are indices 0, 1, 2, 3, 4. If `TOP = 4`, which of the following statements is true?

(A) When `TOP = 0`, the stack is completely full and no further elements can be removed (POP).

(B) A PUSH operation always succeeds on an array-based stack regardless of the current stack size.

(C) When `TOP = MAX_SIZE`, the stack is empty and attempting a POP causes stack underflow.

(D) With `MAX_SIZE = 5` and `TOP = 4` (0-indexed), the stack is full; a PUSH causes stack overflow.

Q10. Which of the following correctly describes the behaviour of a *priority queue*?

(A) A priority queue always dequeues the element that was inserted first, following strict FIFO order.

(B) In a priority queue, the element with the highest priority is always dequeued first, regardless of insertion order.

(C) A priority queue can only be implemented using a singly linked list, not arrays or binary heaps.

(D) All elements in a priority queue must be assigned the same priority value to maintain a valid queue.

Q11. Which of the following correctly differentiates between LAN, WAN, and MAN in terms of geographic coverage?

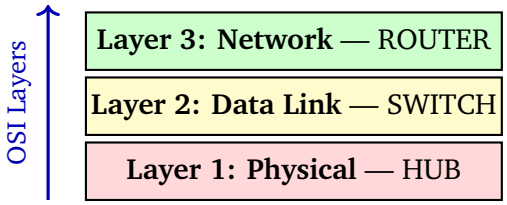
(A) A WAN (Wide Area Network) is limited to a single building or campus environment.

(B) A LAN (Local Area Network) spans entire cities or countries, connecting geographically remote locations.

(C) A MAN (Metropolitan Area Network) covers a city or metropolitan area — larger than a LAN but smaller than a WAN.

(D) LAN and WAN cover exactly the same geographic scope and are used as interchangeable terms.

Q12. The diagram shows three common network devices placed at different layers of the OSI model. Which statement correctly maps each device to its operating layer?



(A) Hub operates at Layer 1 (Physical), Switch at Layer 2 (Data Link), and Router at Layer 3 (Network).

(B) Hub operates at Layer 2 (Data Link), Switch at Layer 3 (Network), and Router at Layer 1 (Physical).

(C) Hub operates at Layer 3 (Network), Switch at Layer 1 (Physical), and Router at Layer 2 (Data Link).

(D) All three devices — Hub, Switch, and Router — operate at Layer 2 (Data Link) of the OSI model.

Q13. In SQL, what is the key difference between an `INNER JOIN` and a `LEFT JOIN`?

(A) `INNER JOIN` returns all rows from both tables, including rows with no matching values in the other table.

(B) `LEFT JOIN` returns only rows where matching values exist in both the left and right tables.

(C) `INNER JOIN` returns all rows from the left table along with only matched rows from the right table.

(D) `INNER JOIN` returns only matched rows from both tables; `LEFT JOIN` returns all rows from the left table plus matched rows from the right.

Q14. Third Normal Form (3NF) is defined in terms of which of the following conditions?

(A) 3NF requires only that every non-key attribute is partially dependent on some candidate key.

(B) 3NF requires 2NF plus the elimination of transitive dependencies — every non-key attribute must depend only on the primary key.

(C) 3NF requires that no repeating groups or multivalued attributes exist in any relation in the database.

(D) 3NF mandates that every attribute be multivalued and determined by a composite primary key.

Q15. Which of the following correctly categorises SQL commands into DDL (Data Definition Language) and DML (Data Manipulation Language)?

(A) INSERT, UPDATE, and DELETE are DDL commands that define and modify the database structure.

(B) CREATE, ALTER, and DROP are DML commands used to manipulate the data stored in a database.

(C) DDL commands include CREATE, ALTER, and DROP; DML commands include INSERT, UPDATE, DELETE, and SELECT.

(D) SELECT is a DDL command and CREATE is a DML command in standard SQL syntax.

Q16. Which of the following is the most accurate description of *recursion* in programming?

(A) Recursion is a technique where a function calls itself; a base case is mandatory to prevent infinite recursion.

(B) Recursion occurs when two separate functions alternately call each other; no base case is required.

(C) A recursive function is permitted to call itself at most once during any single execution path.

(D) Recursion is supported only in purely object-oriented programming languages, not in procedural ones.

Q17. Which of the following correctly describes the *binary search* algorithm and its requirements?

(A) Binary search works on any unsorted list and has a worst-case time complexity of $O(n)$.

(B) Binary search requires an unsorted array and begins comparisons from the last element inward.

(C) Binary search has a worst-case time complexity of $O(n \log n)$ even when applied to a sorted array.

(D) Binary search requires a sorted array and has a worst-case time complexity of $O(\log n)$.

Q18. Which of the following is a defining characteristic of *structured programming*?

(A) Structured programming relies on GOTO statements as the primary mechanism for controlling program flow.

(B) Structured programming avoids GOTO statements and uses sequence, selection (if/else), and iteration (loops) as its three basic control structures.

(C) Structured programming mandates the exclusive use of recursion and prohibits all forms of iterative loops.

(D) Structured programming requires that every program module include at least one GOTO for modularity.

Q19. In operating system memory management, which of the following is true about the *paging* technique?

(A) Paging completely eliminates both internal fragmentation and external fragmentation in memory.

(B) Paging causes external fragmentation because page frames of different sizes are scattered in physical memory.

(C) Paging eliminates external fragmentation but may cause internal fragmentation when the last page of a process is not fully utilised.

(D) Paging has no effect on memory fragmentation; fragmentation is handled exclusively by memory compaction.

Q20. Which type of memory is used in USB flash drives, and what is its key characteristic regarding data retention?

(A) USB flash drives use NAND flash memory, which is non-volatile and retains data even without a power supply.

(B) USB flash drives use DRAM (Dynamic RAM), which loses all stored data immediately when power is disconnected.

(C) USB flash drives store data on magnetic platters using the same operating principle as traditional hard disk drives.

(D) USB flash drives use NOR flash memory, which requires a constant power supply to retain stored data.

Q21. Which of the following correctly describes Wi-Fi 6 (IEEE 802.11ax) in terms of frequency bands and theoretical maximum speed?

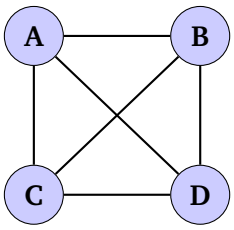
(A) Wi-Fi 6 (802.11ax) operates exclusively on the 5 GHz band with a maximum theoretical speed of 1.3 Gbps.

(B) Wi-Fi 6 (802.11ax) operates only on the 2.4 GHz band and is theoretically limited to a speed of 600 Mbps.

(C) Wi-Fi 6 (802.11ax) entirely replaces the 5 GHz band and operates exclusively on the new 6 GHz spectrum.

(D) Wi-Fi 6 (802.11ax) operates on both 2.4 GHz and 5 GHz bands with a theoretical maximum speed of approximately 9.6 Gbps.

Q22. The figure below shows a complete graph K_4 with 4 vertices. Using the formula $n(n-1)/2$, how many edges does a complete undirected graph with 4 vertices have?



K_4: $n(n-1)/2 = 4 \times 3/2$

(A) A complete undirected graph K_4 with 4 vertices has exactly 4 edges.

(B) A complete undirected graph K_4 with 4 vertices has 6 edges, since $n(n-1)/2 = 4 \times 3/2 = 6$.

(C) A complete undirected graph K_4 with 4 vertices has exactly 8 edges.

(D) A complete undirected graph K_4 with 4 vertices has exactly 12 edges.

Q23. What is the 8-bit two's complement representation of the decimal number -5?

(A) 11111010_2

(B) 11111100_2

(C) 11111011_2

(D) 10000101_2

Q24. In cloud computing, which service model is correctly described by the phrase "software applications delivered over the internet on a subscription basis"?

(A) SaaS (Software as a Service) delivers software applications over the internet on a subscription basis; examples include Gmail and Salesforce.

(B) SaaS (Software as a Service) provides virtualised computing infrastructure such as servers, storage, and networking to clients.

(C) SaaS provides a cloud-based development platform and tools for building, testing, and deploying applications.

(D) SaaS refers to the management of physical data-centre infrastructure owned and operated by cloud service providers.

Q25. Which of the following most accurately describes *open-source software*?

(A) Open-source software makes its source code publicly viewable but strictly prohibits any modification or redistribution by users.

(B) Open-source software requires a commercial licence fee for any form of distribution or modification.

(C) Open-source software is accessible only to registered developers and cannot be freely shared with the general public.

(D) Open-source software has freely available source code that can be studied, modified, and redistributed; examples include Linux and LibreOffice.

Detailed Solutions

Q1.

Solution

Concept — CPU Architecture: The Central Processing Unit (CPU) contains three main functional units: the Control Unit (CU), the Arithmetic Logic Unit (ALU), and Registers. Each has a distinct, non-interchangeable role in executing instructions.

Step 1 — Role of the Control Unit: The Control Unit (CU) fetches instructions from memory, decodes them to determine what operation is needed, and generates control signals to coordinate all other CPU components. It does NOT perform arithmetic or logic operations itself.

Step 2 — Role of the ALU and Registers: The ALU performs all arithmetic operations (add, subtract, multiply) and logic operations (AND, OR, NOT, XOR). Registers are small, ultra-fast temporary storage locations inside the CPU used to hold operands, intermediate results, and addresses during computation.

Why other options are wrong:

- Option A: Registers are temporary, volatile storage — they do not store data permanently. Permanent storage is the role of secondary storage (HDD/SSD).
- Option B: The Control Unit coordinates and decodes instructions; all arithmetic and logical computations are performed by the ALU.
- Option C: Registers are general-purpose storage for any CPU operation; they are not limited to I/O coordination.
- Option D: This correctly describes the Control Unit — it decodes instructions and issues control signals to coordinate all CPU operations.

Final Answer: The Control Unit decodes instructions and coordinates all CPU operations $\Rightarrow$ D

Answer: (D) Go Back to Q 1

Q2.

Solution

Concept — Flatbed Scanner Technology: A flatbed scanner is an input device that converts physical documents and images into digital data using optical sensing technology. The scanning head moves across the document, capturing image data row by row.

Step 1 — CCD Mechanism: Inside a flatbed scanner, the scanning head contains CCD (Charge-Coupled Device) sensors — a linear array of light-sensitive elements. The document is illuminated by a built-in lamp, and reflected light passes through lenses to strike the CCD array.

Step 2 — Digital Conversion: Each CCD element converts the intensity of reflected light into an electrical charge. An Analogue-to-Digital Converter (ADC) translates these charges into digital pixel values. The scanning head advances one row at a time, building the complete digital image.

Why other options are wrong:

- Option A: Thermal sensors detect heat, not reflected light; they are used in thermal printers and barcode readers, not flatbed scanners.
- Option B: CCD sensors digitising images row by row correctly describes the flatbed scanner mechanism.
- Option C: Laser beams are used in laser printers and laser barcode scanners; standard flatbed scanners use CCD or CIS optical sensors.
- Option D: Ultrasonic technology is used for distance measurement and medical sonography, not document scanning.

Final Answer: Flatbed scanners use CCD sensors to digitise images row by row $\Rightarrow$ $\boxed{\text{B}}$

Answer: (B) Go Back to Q 2

Q3.

Solution

Concept — SRAM vs DRAM: Both SRAM and DRAM are types of volatile semiconductor memory, but they differ fundamentally in speed, cost, density, and application.

Step 1 — SRAM Characteristics: SRAM (Static RAM) stores each bit using a flip-flop circuit (6 transistors per cell). It retains data as long as power is supplied

without any refresh cycles. SRAM is significantly faster, but more expensive and less dense than DRAM. It is used for CPU cache (L1, L2, L3 cache).

Step 2 — DRAM Characteristics: DRAM (Dynamic RAM) stores each bit as a charge in a capacitor (1 transistor + 1 capacitor per cell). Because capacitors leak charge, DRAM must be periodically refreshed (thousands of times per second). DRAM is slower and cheaper than SRAM, with much higher density; it is used as main memory (RAM modules on the motherboard).

Why other options are wrong:

- Option A: DRAM is slower (not faster) than SRAM. CPU cache uses SRAM precisely because SRAM's speed matches the processor's requirements.
- Option B: It is DRAM that requires periodic refresh cycles due to capacitor charge leakage; SRAM retains data without any refresh.
- Option C: SRAM is faster, needs no refresh, and is used in cache memory — this correctly states all key distinctions.
- Option D: Only DRAM requires refresh cycles; SRAM is static and does not need refreshing.

Final Answer: SRAM is faster, needs no refresh, and is used in cache memory $\Rightarrow$ C

Answer: (C) Go Back to Q 3

Q4.

Solution

Concept — BIOS (Basic Input/Output System): BIOS is system firmware — a low-level software program permanently embedded into a chip on the computer motherboard. It is the first software that runs when a computer is powered on, responsible for hardware initialisation before the operating system loads.

Step 1 — BIOS Boot Sequence (POST): When the computer powers on, BIOS executes a POST (Power-On Self-Test) to detect and verify hardware components: RAM, CPU, keyboard, and storage devices. It then identifies the bootable device (HDD, SSD, USB) and hands control to the OS bootloader.

Step 2 — Storage in ROM/Flash: BIOS is stored in non-volatile ROM (Read-Only Memory) or EEPROM/flash memory on the motherboard, so it persists when power is removed. Modern systems use UEFI (Unified Extensible Firmware Interface), which is the successor to traditional BIOS.

Why other options are wrong:

- Option A: BIOS is firmware stored in ROM that initialises hardware during boot — this is the correct description.
- Option B: BIOS is not stored in RAM (which is volatile and erased on power-off) and is not a general application; it is firmware with a specific system-initialisation role.
- Option C: BIOS resides on a ROM chip on the motherboard, not on the hard disk; the OS kernel is stored on the hard disk.
- Option D: BIOS is not a cache; it is executable firmware code that controls the power-on sequence and hardware detection.

Final Answer: BIOS is firmware stored in ROM that initialises hardware during the boot process $\Rightarrow$ A

Answer: (A) Go Back to Q 4

Q5.

Solution

Concept — Memory Hierarchy: Computer memory is organised as a hierarchy based on two inversely related properties: access speed and storage capacity. Moving up the hierarchy, memory becomes faster, smaller, and more expensive per bit.

Step 1 — Hierarchy Order (fastest to slowest / smallest to largest):

- **Registers** — fastest, smallest, located inside the CPU (bytes to kilobytes)
- **Cache** — L1/L2/L3, very fast, small (kilobytes to megabytes)
- **RAM** — main memory, moderate speed (gigabytes)
- **HDD/SSD** — secondary storage, slowest access, largest capacity (hundreds of GB to terabytes)

Step 2 — The Core Trade-off: Registers are the fastest memory in the system but can hold only a handful of values. Secondary storage (HDD/SSD) holds terabytes but has access times millions of times slower than registers. Cost per bit also decreases as capacity increases down the hierarchy.

Why other options are wrong:

- Option A: RAM is neither the fastest (registers and cache are faster) nor the largest (secondary storage holds far more data).
- Option B: Cache is faster than RAM, not slower, and cache is much smaller in capacity than RAM.

- Option C: Secondary storage is slower than registers (not faster), and it has much larger capacity, not smaller.
- Option D: Registers are the fastest and smallest; HDD/SSD is the largest and slowest — this correctly describes both ends of the hierarchy.

Final Answer: Registers are fastest and smallest; secondary storage is largest and slowest $\Rightarrow$ D

Answer: (D) Go Back to Q 5

Q6.

Solution

Concept — Deadlock and the Four Coffman Conditions: A deadlock is a state where two or more processes are permanently blocked, each waiting for a resource held by another. In 1971, Coffman, Elphick, and Shoshani identified four conditions that must *all* hold simultaneously for deadlock to occur.

Step 1 — The Four Coffman Conditions:

(a) **Mutual Exclusion:** Only one process may use a resource at a time; resources cannot be shared.
(b) **Hold and Wait:** A process holding at least one resource waits to acquire additional resources held by others.
(c) **No Preemption:** Resources cannot be forcibly taken away; a process must release them voluntarily.
(d) **Circular Wait:** A closed chain of processes exists where P1 waits for a resource held by P2, P2 waits for P3, . . . , Pn waits for P1.

Step 2 — Why Circular Wait is Key: Circular Wait creates the closed dependency loop that prevents any process in the chain from making progress. Breaking any one of the four conditions is sufficient to prevent deadlock.

Why other options are wrong:

- Option A: Deadlock requires at least two processes forming a circular dependency; a single process cannot create a circular wait with itself.
- Option B: Circular Wait is one of the four Coffman conditions — this is the correct answer.
- Option C: Operating systems do not always resolve deadlocks automatically; many use deadlock avoidance (Banker's Algorithm) or detection-and-recovery strategies.

- Option D: Mutual Exclusion IS one of the four Coffman conditions; it is required for deadlock to occur.

Final Answer: Circular Wait is one of the four Coffman conditions necessary for deadlock $\Rightarrow$ B

Answer: (B) Go Back to Q 6

Q7.

Solution

Concept — Binary Subtraction: Binary subtraction follows the same column-by-column process as decimal subtraction. The borrowing rule in binary: $0 - 1$ requires borrowing 1 from the next higher bit, making the current bit $10_2 - 1_2 = 1$ with a borrow of 1 from the left.

Step 1 — Perform Binary Subtraction column by column (right to left):

$$\begin{array}{r} 1101_2 \\ -\ 0101_2 \\ \hline \end{array}$$

- Bit 0 (rightmost): $1 - 1 = 0$
- Bit 1: $0 - 0 = 0$
- Bit 2: $1 - 1 = 0$
- Bit 3 (leftmost): $1 - 0 = 1$

Result: $\mathbf{1000_2}$

Step 2 — Verify in Decimal: $1101_2 = 8 + 4 + 0 + 1 = 13$; $0101_2 = 4 + 1 = 5$; $13 - 5 = 8$. And $1000_2 = 8$. ✓

Why other options are wrong:

- Option A: $0111_2 = 7$; this would imply $13 - 5 = 7$, which is arithmetically incorrect.
- Option B: $1010_2 = 10$; this would imply $13 - 5 = 10$, which is incorrect.
- Option C: $1000_2 = 8$ and $13 - 5 = 8$ — this is the correct result.
- Option D: $0110_2 = 6$; this would imply $13 - 5 = 6$, which is incorrect.

Final Answer: $1101_2 - 0101_2 = 1000_2 = 8$ in decimal $\Rightarrow$ C

Answer: (C) Go Back to Q 7

Q8.

Solution

Concept — Universal Logic Gates: A universal gate is a logic gate from which any Boolean function (AND, OR, NOT, XOR, etc.) can be built using only that single gate type, without needing any other type of gate. Both NAND and NOR are universal gates.

Step 1 — Building NOT from NAND: Connect both inputs of a NAND gate to the same input A:

$$\text{NAND}(A, A) = \overline{A \cdot A} = \overline{A}$$

This implements a NOT (inverter) gate using only one NAND gate.

Step 2 — Building AND and OR from NAND:

- **AND:** $A \cdot B = \overline{\overline{A \cdot B}} = \text{NOT}(\text{NAND}(A, B))$ — invert the NAND output with another NAND used as NOT.
- **OR:** $A + B = \overline{\overline{A} \cdot \overline{B}} = \text{NAND}(\overline{A}, \overline{B})$ — apply NOT to each input (via NAND-as-NOT), then feed into NAND.

Since NOT, AND, and OR can all be built from NAND, any Boolean expression is implementable with NAND gates alone.

Why other options are wrong:

- Option A: NAND is universal — any Boolean function can be built using only NAND gates. This is correct.
- Option B: AND is not universal; you cannot build a NOT gate from AND gates alone (AND never produces inversion without other gate types).
- Option C: OR is not universal by itself; a NOT gate cannot be created from OR gates alone.
- Option D: XOR is not universal; while XOR has useful properties (parity checking, half-adders), it cannot alone implement all Boolean functions without additional gate types.

Final Answer: NAND is a universal gate — all Boolean functions can be built using only NAND gates $\Rightarrow$ A

Answer: (A) Go Back to Q 8

Q9.

Solution

Concept — Stack Overflow in Array Implementation: In an array-based stack with 0-based indexing and `MAX_SIZE = 5`, valid index positions are 0, 1, 2, 3, and 4. The variable `TOP` points to the index of the most recently pushed element. An empty stack is indicated by `TOP = -1`.

Step 1 — Determine Stack State when TOP = 4: With 0-based indexing and `MAX_SIZE = 5`, the maximum valid index is $5 - 1 = 4$. When `TOP = 4`, all five positions (indices 0 through 4) are occupied. The stack is **full**.

Step 2 — Effect of a PUSH Operation: Attempting PUSH when `TOP = 4` would increment `TOP` to 5. Since index 5 is beyond the array bound (`MAX_SIZE = 5`, valid indices: 0–4), this results in a **stack overflow** error. The PUSH must be rejected.

Why other options are wrong:

- Option A: `TOP = 0` means only index 0 is occupied (one element in the stack); the stack is not full, and POP is valid. An empty stack is indicated by `TOP = -1`.
- Option B: PUSH does not always succeed; on a full array-based stack (`TOP = MAX_SIZE - 1`), PUSH causes stack overflow.
- Option C: `TOP = MAX_SIZE` (i.e., TOP = 5) represents an overflow condition, not an empty state. An empty stack has `TOP = -1`.
- Option D: With `MAX_SIZE = 5` and `TOP = 4`, all 5 slots (0–4) are used; the stack is full, and the next PUSH causes stack overflow — this is correct.

Final Answer: TOP = 4 with MAX_SIZE = 5 (0-indexed) means the stack is full; PUSH causes stack overflow $\Rightarrow$ D

Answer: (D) Go Back to Q 9

Q10.

Solution

Concept — Priority Queue: A priority queue is an abstract data type in which each element has an associated priority value. Unlike a standard queue (which follows FIFO), a priority queue always serves the element with the highest priority first, regardless of when it was inserted.

Step 1 — Dequeue Order: When an element is dequeued (removed), the element with the highest priority is selected, not the oldest. If two elements share the same

priority, FIFO order is applied among them.

Step 2 — Implementations: Priority queues are most efficiently implemented using a binary heap (min-heap for lowest-priority-first, max-heap for highest-priority-first), achieving $O(\log n)$ insertion and deletion. They can also be implemented using sorted arrays, sorted linked lists, or Fibonacci heaps.

Why other options are wrong:

- Option A: Strict FIFO (oldest element dequeued first) describes a regular queue, not a priority queue. Priority overrides insertion order.
- Option B: Dequeuing the element with the highest priority first, regardless of insertion order, is the defining property of a priority queue — this is correct.
- Option C: Priority queues can be implemented using arrays, heaps, or linked lists. No single implementation is mandated.
- Option D: If all elements had equal priority, a priority queue would degenerate into a FIFO queue. Varying priorities are the defining use case.

Final Answer: A priority queue dequeues the element with the highest priority first, regardless of insertion order $\Rightarrow$ $\boxed{\text{B}}$

Answer: (B) Go Back to Q 10

Q11.

Solution

Concept — Network Types by Geographic Scope: Computer networks are classified by the geographic area they cover. The three primary categories are LAN, MAN, and WAN.

Step 1 — Geographic Coverage Comparison:

- **LAN (Local Area Network):** Covers a limited area such as a single room, floor, building, or campus. Typical range: up to a few kilometres. High speed (100 Mbps to 10 Gbps). Example: an office or university campus network.
- **MAN (Metropolitan Area Network):** Covers a city or metropolitan area, typically 5 to 50 km. Bridges multiple LANs across a city. Example: a city-wide cable TV network or municipal broadband.
- **WAN (Wide Area Network):** Spans large geographic regions — countries or continents. Lower speed than LAN relative to distance. Example: the internet, or leased-line connections between cities.

Step 2 — Size Order: LAN $<$ MAN $<$ WAN in terms of geographic coverage area.

Why other options are wrong:

- Option A: A WAN spans large regions (countries/continents); a single building is covered by a LAN.
- Option B: A LAN is restricted to a local area; spanning cities or countries is the role of a WAN.
- Option C: MAN covers a city/metropolitan area, larger than a LAN but smaller than a WAN — this is correct.
- Option D: LAN and WAN have very different geographic scopes; they are not interchangeable.

Final Answer: MAN covers a city/metropolitan area — larger than LAN, smaller than WAN $\Rightarrow$ C

Answer: (C) Go Back to Q 11

Q12.

Solution

Concept — Network Devices and OSI Layers: Different network devices operate at different layers of the OSI (Open Systems Interconnection) reference model. The layer at which a device operates determines what type of addressing it uses and what intelligence it applies to forwarding traffic.

Step 1 — Hub at Layer 1 (Physical): A hub is a simple repeater. It receives an incoming electrical signal and broadcasts it to all other ports without examining the data content. Since it operates purely on electrical signals and has no concept of frames or packets, it works at Layer 1 (Physical).

Step 2 — Switch at Layer 2 and Router at Layer 3:

- **Switch $\rightarrow$ Layer 2 (Data Link):** A switch reads MAC (Media Access Control) addresses in Ethernet frames to forward traffic only to the correct port. MAC addresses are a Layer 2 concept.
- **Router $\rightarrow$ Layer 3 (Network):** A router reads IP addresses to make forwarding decisions and route packets between different networks. IP addresses are a Layer 3 concept.

Why other options are wrong:

- Option A: Hub at L1, Switch at L2, Router at L3 — this is the standard, correct mapping.

- Option B: Hub cannot read MAC addresses (Layer 2 concept); it has no frame intelligence and works only at Layer 1.
- Option C: Hub cannot route packets based on IP addresses (Layer 3 concept); routers do that.
- Option D: Only switches work primarily at Layer 2. Hubs are Layer 1 and routers are Layer 3 — they do not all share Layer 2.

Final Answer: Hub at Layer 1 (Physical), Switch at Layer 2 (Data Link), Router at Layer 3 (Network) $\Rightarrow$ A

Answer: (A) Go Back to Q 12

Q13.

Solution

Concept — SQL JOIN Types: SQL JOINs combine rows from two or more tables based on a related column condition. The type of JOIN determines which rows are included in the result when there is no match in one of the tables.

Step 1 — INNER JOIN: Returns only rows where the join condition is satisfied in *both* tables simultaneously. If a row in Table A has no matching row in Table B (or vice versa), that row is completely excluded from the result set.

Step 2 — LEFT JOIN (LEFT OUTER JOIN): Returns *all* rows from the *left* table, plus the matching rows from the right table. Where no match exists in the right table, the right-table columns are filled with `NULL`. No rows from the left table are ever dropped.

Why other options are wrong:

- Option A: INNER JOIN does not return unmatched rows; unmatched rows are only included by OUTER JOINs (LEFT, RIGHT, or FULL OUTER JOIN).
- Option B: Returning only matched rows from both tables describes INNER JOIN, not LEFT JOIN.
- Option C: Returning all left-table rows plus matched right-table rows describes LEFT JOIN, not INNER JOIN.
- Option D: INNER JOIN $\rightarrow$ matched rows only; LEFT JOIN $\rightarrow$ all left rows + matched right rows (NULLs for unmatched right) — this is the correct distinction.

Final Answer: INNER JOIN returns only matched rows; LEFT JOIN returns all left-table rows plus matched right-table rows $\Rightarrow$ D

Answer: (D) Go Back to Q 13

Q14.

Solution

Concept — Database Normalisation — 3NF: Normalisation is the process of organising a relational database to reduce redundancy and improve data integrity. Each Normal Form (NF) adds additional constraints on top of the previous form.

Step 1 — Prerequisites: 1NF and 2NF:

- **1NF:** All attributes contain atomic (indivisible) values; no repeating groups.
- **2NF:** Must be in 1NF, plus every non-key attribute is *fully* functionally dependent on the entire primary key (no partial dependencies; relevant only for composite keys).

Step 2 — 3NF Definition — Eliminating Transitive Dependencies: A relation is in 3NF if it is in 2NF and no non-key attribute transitively depends on the primary key through another non-key attribute. Formally: for every functional dependency $X \to Y$ in the relation, either X is a superkey or Y is a prime attribute (part of some candidate key). In practice: every non-key attribute must depend *directly and only* on the primary key.

Why other options are wrong:

- Option A: Partial dependency (a non-key attribute depending on only part of a composite primary key) is eliminated by 2NF, not 3NF.
- Option B: 3NF = 2NF + no transitive dependencies; non-key attributes depend only on the primary key — this is the correct definition.
- Option C: Repeating groups are addressed by 1NF (the most basic normal form), not 3NF.
- Option D: Multivalued dependencies are the concern of 4NF; BCNF (Boyce-Codd Normal Form) deals with stricter superkey requirements, not 3NF directly.

Final Answer: 3NF requires 2NF plus the elimination of all transitive dependencies $\Rightarrow$ $\boxed{\text{B}}$

Answer: (B) Go Back to Q 14

Q15.

Solution

Concept — SQL Command Categories (DDL and DML): SQL commands are grouped by their purpose. DDL (Data Definition Language) commands define and modify the database structure (schema), while DML (Data Manipulation Language) commands manipulate the data stored within that structure.

Step 1 — DDL (Data Definition Language) Commands: DDL commands operate on database objects (tables, views, indexes):

- `CREATE` — creates a new table, view, index, or database
- `ALTER` — modifies an existing table (add/drop/rename columns, change data types)
- `DROP` — permanently removes a table or other database object
- `TRUNCATE` — removes all rows from a table but keeps the table structure

Step 2 — DML (Data Manipulation Language) Commands: DML commands operate on data rows within existing database objects:

- `INSERT` — adds new rows to a table
- `UPDATE` — modifies values in existing rows
- `DELETE` — removes specific rows from a table
- `SELECT` — retrieves (reads) rows from one or more tables

Why other options are wrong:

- Option A: INSERT, UPDATE, and DELETE are DML commands (they manipulate data rows), not DDL commands.
- Option B: CREATE, ALTER, and DROP are DDL commands (they define/modify schema), not DML commands.
- Option C: DDL = CREATE/ALTER/DROP and DML = INSERT/UPDATE/DELETE/SELECT — this correctly categorises all listed commands.
- Option D: SELECT is a DML command (it retrieves data) and CREATE is a DDL command; this option has them swapped.

Final Answer: DDL: CREATE, ALTER, DROP; DML: INSERT, UPDATE, DELETE, SELECT $\Rightarrow$ C

Answer: (C) Go Back to Q 15

Q16.

Solution

Concept — Recursion in Programming: Recursion is a programming technique where a function solves a problem by calling itself with a simpler or smaller version of the original problem. Each recursive call reduces the problem until a base case is reached.

Step 1 — Two Essential Components:

- **Recursive Case:** The function calls itself with a modified argument (typically smaller). Example: `factorial(n) = n` $\times$ `factorial(n−1)` for $n > 0$.
- **Base Case:** A condition that terminates the recursion without making another call. Example: `factorial(0) = 1`. Without a base case, the function recurses infinitely, exhausting the call stack and causing a stack overflow.

Step 2 — Call Stack Mechanics: Each recursive call is pushed onto the call stack as a new stack frame. When the base case is reached, the calls unwind (pop off the stack), with each returning its result upward to the caller.

Why other options are wrong:

- Option A: A function calling itself with a mandatory base case is the precise definition of recursion — this is correct.
- Option B: Two functions calling each other is *mutual* (indirect) recursion, a special case. A base case is always required to prevent infinite mutual calls.
- Option C: Many recursive algorithms make multiple self-calls per invocation (e.g., binary tree traversal, merge sort). The "at most once" restriction does not define recursion.
- Option D: Recursion is supported in many procedural languages (C, Pascal, FORTRAN) and is not exclusive to object-oriented languages.

Final Answer: Recursion = function calls itself; a base case is mandatory to prevent infinite recursion $\Rightarrow$ $\boxed{\text{A}}$

Answer: (A) Go Back to Q 16

Q17.

Solution

Concept — Binary Search Algorithm: Binary search is an efficient search algorithm that operates on a *sorted* array by repeatedly halving the search space. It is a classic application of the divide-and-conquer strategy.

Step 1 — Prerequisite and Mechanism: Binary search requires the input array to be sorted in ascending (or descending) order. It works by comparing the target value to the middle element:

- If target $=$ middle: found.
- If target $<$ middle: search the left half.
- If target $>$ middle: search the right half.

Step 2 — Time Complexity: At each step, the search range is halved: $n \to n/2 \to n/4 \to \cdots \to 1$. The number of steps in the worst case is $\lceil \log_2 n \rceil$, yielding a worst-case time complexity of $O(\log n)$.

$$\text{Example: For } n = 1024, \text{ at most } \log_2 1024 = 10 \text{ comparisons needed.}$$

Why other options are wrong:

- Option A: Binary search does NOT work on unsorted arrays; the halving decision is only valid when data is sorted. $O(n)$ is the complexity of linear (sequential) search.
- Option B: Binary search always requires a sorted array. Searching from the last element inward describes a reverse linear scan, not binary search.
- Option C: $O(n \log n)$ is the time complexity of sorting algorithms such as merge sort and heap sort, not of binary search.
- Option D: Sorted array required + worst-case $O(\log n)$ — this precisely and correctly describes binary search.

Final Answer: Binary search requires a sorted array; worst-case time complexity is $O(\log n) \Rightarrow \boxed{\text{D}}$

Answer: (D) Go Back to Q 17

Q18.

Solution

Concept — Structured Programming: Structured programming is a programming paradigm advocated by Edsger Dijkstra (notably in his 1968 letter "Go To Statement Considered Harmful") that improves program clarity, quality, and development time by restricting flow of control to three fundamental constructs.

Step 1 — The Three Control Structures (Böhm-Jacopini Theorem): Any algorithm can be expressed using exactly three logical constructs:

- **Sequence:** Execute statements one after another in order.
- **Selection:** Make decisions using `if/else` or `switch/case` statements.
- **Iteration:** Repeat actions using `for`, `while`, or `do-while` loops.

Step 2 — Avoidance of GOTO: GOTO statements create unstructured, unpredictable control flow ("spaghetti code"), making programs difficult to read, test, and maintain. Structured programming explicitly prohibits or discourages GOTO in favour of the three structured constructs above.

Why other options are wrong:

- Option A: Structured programming was specifically designed to *eliminate* GOTO statements, not to use them as the primary mechanism.
- Option B: No GOTO + sequence, selection, and iteration as the only control structures is the exact definition of structured programming — this is correct.
- Option C: Structured programming encourages iteration (loops are one of the three fundamental constructs). It does not prohibit loops or mandate recursion exclusively.
- Option D: Requiring GOTO in every module directly contradicts the founding principle of structured programming.

Final Answer: Structured programming avoids GOTO and uses sequence, selection, and iteration $\Rightarrow$ $\boxed{\text{B}}$

Answer: (B) Go Back to Q 18

Q19.

Solution

Concept — OS Paging and Memory Fragmentation: Paging is a memory management technique that divides physical memory into fixed-size blocks called *frames* and logical memory (process address space) into fixed-size blocks called *pages*. Pages are loaded into any available frame, not necessarily contiguous.

Step 1 — Eliminating External Fragmentation: External fragmentation occurs when free memory is scattered in small gaps between allocated blocks, making it impossible to satisfy large allocation requests despite enough total free memory. Because all frames are the same fixed size, any free frame can hold any page. No unusable gaps form between frames, so paging **eliminates external fragmentation**.

Step 2 — Causing Internal Fragmentation: A process's last page often does not completely fill its assigned frame. For example, if the frame size is 4 KB and a process's last page uses only 1.5 KB, the remaining 2.5 KB in that frame is wasted. This wasted space *inside* an allocated frame is **internal fragmentation**.

Why other options are wrong:

- Option A: Paging does not eliminate internal fragmentation. The last page of a process frequently does not fully occupy its frame.
- Option B: Paging eliminates external fragmentation (not causes it). External fragmentation arises in variable-size allocation (segmentation), not in fixed-size paging.
- Option C: Paging eliminates external fragmentation but may cause internal fragmentation in the last page/frame — this is the correct statement.
- Option D: Paging directly reduces external fragmentation by design. Memory compaction is a remedy used with segmentation, not paging.

Final Answer: Paging eliminates external fragmentation but may cause internal fragmentation $\Rightarrow$ C

Answer: (C) Go Back to Q 19

Q20.

Solution

Concept — USB Flash Drive Memory Technology: USB flash drives are compact, portable, solid-state storage devices. Their internal memory type determines their key properties such as data retention, durability, and access characteristics.

Step 1 — NAND Flash Memory: USB flash drives use **NAND flash memory**, a type of non-volatile solid-state memory. "Non-volatile" means that data is permanently retained even when the power supply is completely removed. NAND flash stores each bit using floating-gate MOSFET transistors that trap electrical charge; no power is needed to maintain the charge state.

Step 2 — Key Properties of NAND Flash:

- **Non-volatile:** Data persists without power (unlike DRAM, which is volatile).
- **No moving parts:** Silent, shock-resistant (unlike HDDs with spinning platters).
- **Limited write cycles:** NAND cells wear out after $10{,}000$–$100{,}000$ program/erase cycles (managed by wear levelling).
- **Fast read, slower write:** Reads are fast; writes/erases are slower and block-granular.

Why other options are wrong:

- Option A: NAND flash memory is non-volatile and retains data without power — this correctly describes USB flash drives.
- Option B: DRAM is volatile; all stored data is lost immediately when power is removed. USB flash drives do not use DRAM.
- Option C: Magnetic platters are used in traditional HDDs, which have spinning disks and read/write heads. USB flash drives have no mechanical moving parts.
- Option D: NOR flash is a different type of flash memory used in firmware/BIOS chips and microcontrollers. USB flash drives use NAND flash (higher density, lower cost). Additionally, NOR flash is also non-volatile — it retains data without power — so this option is doubly incorrect.

Final Answer: USB flash drives use NAND flash memory — non-volatile, retains data without a power supply $\Rightarrow$ $\boxed{\text{A}}$

Answer: (A) Go Back to Q 20

Q21.

Solution

Concept — Wi-Fi 6 (IEEE 802.11ax): Wi-Fi 6 is the sixth generation of Wi-Fi technology, standardised by the IEEE as 802.11ax and certified by the Wi-Fi Alliance as "Wi-Fi 6." It was designed primarily to improve efficiency, capacity, and performance in dense environments with many simultaneously connected devices.

Step 1 — Frequency Bands: Wi-Fi 6 (802.11ax) operates on *both* the 2.4 GHz and 5 GHz frequency bands simultaneously. (Wi-Fi 6E, a later extension, added the 6 GHz band, but standard Wi-Fi 6 does not include 6 GHz.)

Step 2 — Theoretical Maximum Speed and Key Technologies: Wi-Fi 6 achieves a theoretical maximum aggregate throughput of approximately **9.6 Gbps**, a significant increase from Wi-Fi 5's ∼3.5 Gbps. Key enabling technologies include:

- **OFDMA** (Orthogonal Frequency Division Multiple Access): splits channels to serve multiple users simultaneously
- **MU-MIMO** (Multi-User MIMO): 8×8 spatial streams for simultaneous multi-device communication
- **BSS Colouring**: reduces co-channel interference in dense environments
- **TWT** (Target Wake Time): reduces battery drain for IoT and mobile devices

Why other options are wrong:

- Option A: Wi-Fi 6 does not operate exclusively on 5 GHz; it uses both 2.4 GHz and 5 GHz. 1.3 Gbps is closer to a single-stream Wi-Fi 5 speed.
- Option B: Wi-Fi 6 does not operate only on 2.4 GHz. 600 Mbps was the approximate maximum for Wi-Fi 4 (802.11n).
- Option C: The 6 GHz band is used by Wi-Fi 6E (the extended variant), not standard Wi-Fi 6. Standard Wi-Fi 6 does not operate on 6 GHz.
- Option D: Both 2.4 GHz and 5 GHz bands with ∼9.6 Gbps theoretical maximum — this correctly describes Wi-Fi 6.

Final Answer: Wi-Fi 6 (802.11ax) uses 2.4 GHz and 5 GHz bands with a theoretical maximum of ∼9.6 Gbps ⇒ D

Answer: (D) Go Back to Q 21

Q22.

Solution

Concept — Complete Graph K_n: A complete graph K_n is an undirected graph in which every pair of distinct vertices is connected by exactly one edge. The total number of edges is given by the combination formula $\binom{n}{2}$.

Step 1 — Apply the Formula for $n = 4$:

$$\text{Number of edges in } K_4 = \binom{4}{2} = \frac{n(n-1)}{2} = \frac{4 \times 3}{2} = \frac{12}{2} = \mathbf{6}$$

Step 2 — Verify by Enumeration: With vertices $\{A, B, C, D\}$, all possible edges are:

$$AB,\ AC,\ AD,\ BC,\ BD,\ CD$$

Counting: $3+2+1 = 6$ edges. ✓ (Vertex A connects to 3 others, B to 2 remaining, C to 1 remaining.)

Why other options are wrong:

- Option A: 4 edges would form a simple cycle C_4 (a square) — not a complete graph. K_4 requires $\frac{4\times3}{2} = 6$ edges.
- Option B: $n(n-1)/2 = 4 \times 3/2 = 6$ edges — this is the correct count for K_4.
- Option C: 8 edges exceeds the maximum possible for 4 vertices in a simple graph (maximum is 6).
- Option D: 12 would be the edge count for a complete *directed* graph (digraph) on 4 vertices (each pair has 2 directed edges: $4 \times 3 = 12$); for an *undirected* complete graph, it is 6.

Final Answer: K_4 has $n(n-1)/2 = 4 \times 3/2 = 6$ edges $\Rightarrow$ B

Answer: (B) Go Back to Q 22

Q23.

Solution

Concept — Two's Complement Representation: Two's complement is the standard method for representing signed integers in binary. For an n-bit system, a negative number $-x$ is represented as the binary of $2^n - x$. Practically, it is computed by inverting all bits (one's complement) and adding 1.

Step 1 — Binary representation of $+5$ in 8 bits:

$$+5 = 00000101_2$$

Step 2 — Two's Complement of -5:

Step 2a (invert all bits — one's complement): $00000101 \longrightarrow 11111010$

Step 2b (add 1): $11111010 + 1 = \mathbf{11111011}$

Therefore, -5 in 8-bit two's complement is 11111011_2.

Verification: 11111011_2 has MSB = 1 (negative). Invert: 00000100. Add 1: $00000101 = 5$. So the value is -5. ✓

Why other options are wrong:

- Option A: 11111010_2 is the one's complement of 5 (after inverting bits, before adding 1). It represents -6 in two's complement, not -5.
- Option B: 11111100_2 in two's complement: invert $\rightarrow 00000011 = 3$; add $1 \rightarrow 00000100 = 4$; so $11111100_2 = -4$, not -5.
- Option C: 11111011_2 is obtained correctly by inverting 00000101 to 11111010 and adding 1 to get 11111011 — this is the two's complement of -5.
- Option D: 10000101_2 is the sign-magnitude representation of -5 (MSB = 1 for negative, magnitude = $00000101 = 5$). Two's complement and sign-magnitude are different encoding schemes.

Final Answer: -5 in 8-bit two's complement: $00000101 \xrightarrow{\text{invert}} 11111010 \xrightarrow{+1} \mathbf{11111011_2} \Rightarrow \boxed{C}$

Answer: (C) Go Back to Q 23

Q24.

Solution

Concept — Cloud Computing Service Models: Cloud computing delivers computing resources over the internet ("the cloud"). There are three primary service models, each providing a different level of abstraction and management responsibility split between the provider and the user.

Step 1 — The Three Cloud Service Models:

- **IaaS (Infrastructure as a Service):** Provides virtualised computing infras-

tructure — virtual machines, storage, and networking. Users manage OS, middleware, and applications. Example: AWS EC2, Microsoft Azure VMs, Google Compute Engine.

- **PaaS (Platform as a Service):** Provides a managed development and deployment platform (runtime environments, databases, dev tools). Users manage applications and data. Example: Heroku, Google App Engine, Azure App Service.
- **SaaS (Software as a Service):** Delivers fully functional software applications over the internet on a subscription basis. Users access via web browser; the provider manages everything. Example: Gmail, Salesforce, Microsoft 365, Dropbox.

Step 2 — Identify the Model: "Software applications delivered over the internet on a subscription basis" matches **SaaS**. In SaaS, the user manages nothing infrastructure-wise — only their data and settings.

Why other options are wrong:

- Option A: SaaS delivers software over the internet on subscription (Gmail, Salesforce) — this is the correct description.
- Option B: Providing virtualised hardware (servers, storage, networking) describes IaaS, not SaaS.
- Option C: Providing a development and deployment platform describes PaaS, not SaaS.
- Option D: Managing physical data-centre hardware is an internal provider function; it is not a named cloud service model delivered to customers.

Final Answer: SaaS delivers software applications over the internet on subscription (e.g., Gmail, Salesforce) $\Rightarrow$ A

Answer: (A) Go Back to Q 24

Q25.

Solution

Concept — Open-Source Software: Open-source software (OSS) is software released under a licence that grants all users the rights to study, use, modify, and distribute the software and its source code freely. The Open Source Initiative (OSI) maintains the formal definition and approves compliant licences.

Step 1 — Key Characteristics of Open-Source Software:

- **Free redistribution:** Anyone may freely give away or sell the software.
- **Source code access:** The source code must be included or freely obtainable.
- **Derived works:** Modifications and derived works are permitted and may be distributed under the same licence.
- **No discrimination:** The licence must not restrict use by any person, group, or field of endeavour.

Step 2 — Prominent Examples:

- **Linux:** Open-source OS kernel; forms the basis of Android, Ubuntu, Fedora, and most server operating systems.
- **LibreOffice:** Open-source office productivity suite; a free alternative to Microsoft Office.
- Other examples: Mozilla Firefox, VLC Media Player, Python, MySQL, Apache HTTP Server, Git.

Why other options are wrong:

- Option A: Open-source explicitly *permits* modification and redistribution. Prohibiting them would make it "source-available" or proprietary software, not open-source.
- Option B: Open-source software does not require a licence fee for distribution or modification; requiring payment contradicts the open-source definition and describes proprietary commercial software.
- Option C: Open-source software is available to *anyone*, not restricted to registered developers. Restricting access describes proprietary or source-available (but not open-source) software.
- Option D: Freely available, modifiable, and redistributable source code with Linux and LibreOffice as canonical examples — this accurately and completely describes open-source software.

Final Answer: Open-source software has freely available source code that can be modified and redistributed (e.g., Linux, LibreOffice) $\Rightarrow$ $\boxed{\text{D}}$

Answer: (D) Go Back to Q 25

Answer Key

Q	Ans	Q	Ans	Q	Ans	Q	Ans	Q	Ans
1	D	2	B	3	C	4	A	5	D
6	B	7	C	8	A	9	D	10	B
11	C	12	A	13	D	14	B	15	C
16	A	17	D	18	B	19	C	20	A
21	D	22	B	23	C	24	A	25	D