

CIE A-Level Computer Science 9618 Paper 1

Practice Paper (Theory Fundamentals) -

ANSWERS

Total: 75 marks · 1 h 30 min · 2 questions · 11 parts

Q	part	Answer	Mark points	marks
1	(a)	+45 is 0010 1101; inverting the bits gives 1101 0010 and adding 1 gives 11010011. The range is -128 to +127.	1 mark: bit pattern 11010011 1 mark: range -128 to +127 (both ends)	2
1	(b)	Unsigned: $9 \times 16 + 12 = 156$. Signed: the most significant bit is 1 so the value is negative, $156 - 256 = -100$.	1 mark: unsigned 156 1 mark: signed -100	2
1	(c)	Total bits = $800 \times 600 \times 24 = 11520000$ bits, which is 1440000 bytes = 1440 KB. A palette image needs 8 bits per pixel, so 480000 bytes = 480 KB, i.e. 3 times smaller, but only 256 colours are available.	1 mark: $800 \times 600 \times 24$ then divide by 8 1 mark: 1440 KB 1 mark: 480 KB and 3 times smaller	3
1	(d)	Bytes per second = $44100 \times 2 \times 2 = 176400$; times 90 s gives 15876000 bytes = 15.14 MB.	1 mark: $44100 \times 2 \times 2 \times 90$ 1 mark: 15876000 bytes / 15.14 MB	2
1	(e)	The encoded form is 20, black, 30, white, 150, grey - 6 values (3 runs x 2) instead of 200, a saving of 97.0%. If colours change constantly, most runs have length 1 and each pixel needs two values, so the file grows.	1 mark: three runs (length + colour) correct 1 mark: 6 values, 97.0% saved 1 mark: when colours alternate often / few repeated pixels, the file grows	3

2	(a)	Serial sends one bit at a time on a single line; parallel sends several bits at once on several lines. Parallel needs more wires and costs more, and bits travelling on different wires arrive at slightly different times (skew), which gets worse with distance. Serial has no skew and is cheaper over distance, so it is used for USB and network cabling.	1 mark: serial one bit at a time / parallel several bits at once 1 mark: parallel suffers skew / needs more wires 1 mark: serial has no skew over distance and is cheaper	3
2	(b)	12 MiB = 100663296 bits; dividing by 20×10^6 gives 5.03 s. The real time is longer because of headers, gaps between packets, acknowledgements and retransmissions, other users sharing the bandwidth and errors on the line.	1 mark: converted to 100663296 bits 1 mark: 5.03 s 1 mark: any overhead (headers / retransmission / shared bandwidth)	3
2	(c)	A MAC address is burned into the NIC by the manufacturer, is 48 bits and identifies hardware inside one network segment; an IP address is assigned by the network, can change, and is used to route packets between networks. IPv4 uses 32 bits = $2^{32} = 4,294,967,296$ addresses; IPv6 uses 128 bits = 2^{128} addresses (about 3.4×10^{38}).	1 mark: MAC is 48 bits, fixed in hardware, local identification 1 mark: IP is changeable and used for routing 1 mark: IPv4 32 bits, 4,294,967,296 addresses 1 mark: IPv6 128 bits	4
3	(a)	Processor (containing the ALU and control unit), main memory (RAM) and input/output devices, connected by buses. A bus is a set of parallel wires that carries data, addresses and control signals between the units so the CPU can read and write memory and peripherals.	1 mark: processor / ALU / CU 1 mark: main memory or I/O 1 mark: bus carries data, address and control signals	3
3	(b)	$2^{20} = 1,048,576$ addresses, i.e. 1,048,576 bytes = 1024 KiB. The address bus is output by the CPU to specify a memory location and its width fixes the addressable space; the data bus is bidirectional and carries the actual instruction or data, and its width sets how many bits move at once.	1 mark: $2^{20} = 1,048,576$ addresses 1 mark: 1024 KiB 1 mark: address unidirectional for location, data bidirectional for content	3
3	(c)	No moving parts, so it is more shock resistant, silent and uses less power; random access is faster so start-up and loading are quicker (also smaller and cooler).	1 mark: no moving parts (shock resistant / silent / low power) 1 mark: faster access	2

3	(d)	An embedded system is a computer built into a larger device to do one dedicated job, usually running firmware from ROM that the user cannot change. A sensor (water level or temperature) converts physical data into a digital signal for the processor; an actuator (motor, inlet valve) turns the processor output into physical movement.	1 mark: dedicated purpose / built in / firmware in ROM 1 mark: sensor converts physical data to digital input 1 mark: actuator converts output to physical action	3
4	(a)	Fetch: the PC value is copied to the MAR and sent on the address bus; the instruction at that address returns on the data bus into the MDR, is copied to the CIR and the PC is incremented. Decode: the control unit splits the CIR contents into opcode and operand. Execute: the ALU performs the operation or memory is read/written and the result goes to a register or memory, updating the status register.	1 mark: PC to MAR to memory to MDR to CIR, PC incremented 1 mark: control unit decodes opcode / operand 1 mark: ALU executes and result written back / status updated	3
4	(b)	Cycles per second = 3.2×10^9 ; dividing by 2.5 gives 1,280,000,000 instructions per second = 1,280 MIPS. $5,000,000 \times 2.5 / (3.2 \times 10^9) = 0.003906 \text{ s} = 3.91 \text{ ms}$.	1 mark: clock / CPI = 1,280,000,000 per second (1,280 MIPS) 1 mark: instructions x CPI / clock 1 mark: 3.91 ms	3
4	(c)	460 hits, so the hit rate is $460/500 = 92\%$. Average = $0.92 \times 10 + 0.08 \times 120 = 18.80 \text{ ns}$. A cache is small fast memory between the CPU and main memory; frequently used instructions and data are copied into it, so most accesses are served quickly instead of waiting for main memory.	1 mark: hit rate 92% 1 mark: average 18.80 ns with working 1 mark: cache holds frequently used data and cuts main-memory latency	3
5	(a)	Memory management (allocating RAM to programs so they do not overwrite each other); process scheduling (deciding which process gets the CPU, e.g. time slicing); file management (directories, read/write, access rights), plus I/O management and the user interface.	1 mark: memory management with example 1 mark: scheduling / multitasking with example 1 mark: file or I/O management with example	3
5	(b)	A compiler translates the whole source into machine/object code once produce an executable that runs fast, but any change means recompiling. An interpreter translates and runs one line at a time, produces no executable file, and stops at the first error showing the line number, which makes debugging easier but execution slower.	1 mark: compiler translates all at once and creates an executable / interpreter runs line by line 1 mark: difference in speed or error reporting 1 mark: interpreter reports	3

			errors immediately during development	
5	(c)	Backup software (scheduled copies to another medium for recovery) and anti-virus software (scans for and removes malware); also disk defragmenter or compression tools.	1 mark: one utility and its purpose 1 mark: a second utility and its purpose	2
6	(a)	Symmetric encryption uses one shared key for both directions: fast, but the key must be delivered securely. Asymmetric encryption uses a key pair - the public key encrypts and the private key decrypts (or signs) - which solves key distribution but is slower. A digital certificate is issued by a trusted certificate authority and binds a public key to a verified identity so the browser can confirm the server is genuine.	1 mark: symmetric one key / asymmetric key pair 1 mark: asymmetric solves key distribution but is slower 1 mark: certificate from a CA binding public key to identity	3
6	(b)	Combinations = $36^8 = 2,821,109,907,456$. Time = $2,821,109,907,456 / 10^9 = 2,821.1 \text{ s} = \text{about } 47.0 \text{ minutes}$. Improvements: allow capitals and symbols to widen the character set, use longer passwords, enforce two-factor authentication, and limit attempts or lock the account.	1 mark: $36^8 = 2,821,109,907,456$ 1 mark: 47.0 minutes 1 mark: two valid measures (longer / wider character set / 2FA / attempt limit)	3
6	(c)	DDoS: many compromised machines (a botnet) flood a server with requests until bandwidth or connections are exhausted and real users cannot get through. Phishing: a fake email or site impersonating a bank or organisation tricks the user into entering credentials or opening a malicious link. A firewall inspects incoming and outgoing packets and blocks unauthorised ports and addresses by rule.	1 mark: DDoS - many machines flooding to deny service 1 mark: phishing - impersonation to steal credentials 1 mark: firewall filters packets / blocks ports by rule	3
7	(a)	The primary key of Student is StudentID; the primary key of ClubMembership is the composite key (StudentID, ClubID), in which StudentID is a foreign key referencing Student. A student joins many clubs and a club has many students, so the relationship is many-to-many, implemented through the ClubMembership link table.	1 mark: primary key StudentID 1 mark: composite primary key and StudentID as foreign key 1 mark: many-to-many relationship	3
7	(b)	1NF: one value per cell - split the repeated items into many OrderLine rows (OrderID, ItemID, Qty, UnitPrice), removing the repeating group. 2NF: remove partial dependency - item name and price depend	1 mark: 1NF - single value per cell, repeated items split out	3

		only on ItemID, so create Item(ItemID, ItemName, UnitPrice). 3NF: remove transitive dependency - customer name and address depend on CustomerID, not on OrderID, so create Customer(CustomerID, Name, Address).	1 mark: 2NF - partial dependency removed (separate item table) 1 mark: 3NF - transitive dependency removed (separate customer table)	
7	(c)	SELECT COUNT(*) AS BigOrders FROM Orders WHERE Total > 500;	1 mark: COUNT(*) with WHERE Total > 500 1 mark: AS BigOrders and valid syntax	2
8	(a)	A linear search may have to check every element: 1,000,000 comparisons. A binary search halves the range each time, so it needs 20 comparisons ($2^{20} = 1,048,576 > 1,000,000$). The precondition is that the data must already be sorted on the search key and direct access to the middle element.	1 mark: linear 1,000,000 1 mark: binary 20 1 mark: data must be sorted (and randomly accessible)	3
8	(b)	After four full passes the array is 1 3 6 7 9; that is 16 comparisons and 5 swaps. The time complexity is $O(n^2)$ - without an early-exit check all comparisons still run even on sorted data.	1 mark: 1 3 6 7 9 1 mark: 16 comparisons, 5 swaps 1 mark: $O(n^2)$	3
8	(c)	All three keys hash to position 4, which is a collision. Resolve it by chaining (a linked list of keys at that slot) or by linear probing (use the next free slot). A stack is last in first out (LIFO) and a queue is first in first out (FIFO).	1 mark: all three at 4, collision identified 1 mark: chaining / linear probing, and LIFO vs FIFO	2