

APOLLO

A1 Computer fundamentals

HL

Computer Science

First assessment 2027

1 questions | mark schemes

1. (a) *Award [2] max*

To store intermediate arithmetic and logic results/hold operand data;
 ...to reduce the number of memory accesses (by holding intermediate results);

Acts as a temporary storage location;
 ...which stores the values and results from arithmetic and logical operations;

Holds data or the results of an operation;
 ...during the processing cycles;

(b) *Award [1] max*

10101100

(c) *Award [2] max*

Award [1] for each correct column (Difference, Borrower)

X	Y	Difference	Borrower
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

(d) *Award [2] max*

Tracks resource usage;
 Monitors and logs the consumption of resources like CPU, memory, and disk by various processes and users;

Assesses system performance;
 Collects data to identify system bottlenecks, under-utilized resources, or areas for potential optimization;

Assigns costs based on usage;
 In shared computing environments, users may be billed based on the resources they consume;

Facilitates auditing;
 Helps in detecting abnormal behaviors or potential security breaches;

(e) *Award [2] max*

Award [1] per point up to [2] max

Offers a range of pre-built components and libraries to accelerate development;
 Developers can concentrate on coding rather than managing infrastructure;
 Cloud hosted to allow multiple team members to collaborate on application development;
 PaaS often comes with built-in tools for development, testing, integration, and deployment;
 PaaS providers often have strong security protocols to protect the application and data;
 PaaS solutions are designed to be scalable, allowing for easy expansion;

(f) *Award [4] max*

Award [1] per point up to [4] max

Pre-translation: Compiled code runs faster because it has been pre-translated into machine code, while interpreted code is slower due to the on-the-fly interpretation;

Runtime overhead: Compiled code is pre-compiled, so there's no runtime overhead associated with translating source code to machine code, whereas an interpreter must parse, interpret, and translate each line or block of code in real time;

Code optimization: Compilers can take advantage of the architecture's specific capabilities (e.g., optimizing for loops, tasks), whereas interpreter does not have the full view of the program;

Resource utilization: Compilers are more efficient with CPU and memory resources during execution, whereas interpreters consume more CPU cycles and memory during execution (due to the overhead of interpreting code line by line);

Cache utilization: Compiled code has better cache utilization due to the compilation optimizations, whereas the interpreter itself adds an extra layer of instructions that need to be processed, leading to cache misses and slower execution.