2.2 PROGRAMMING TECHNIQUES · 2.2.1(C)(D)

Subroutines, parameters & scope — Mark scheme

30 marks · spec 2.2.1(c)(d)

AO key: AO1 = knowledge · AO2 = application · AO3 = programming / judgement. Code answers are indicative — award for correct logic.

Q	ANSWER	AO	MARKS
1(a)	A function returns a value (1); a procedure carries out a task and need not return a value (1). (2)	AO1	2
1(b)	A parameter is the placeholder in the subroutine definition (1); an argument is the actual value passed in at the call (1). (2)	AO1	2
1(c)	Breaking a program into separate subroutines/modules (1); each doing a specific task (easier to write/test/reuse/maintain) (1). (2)	AO1	2

Q	ANSWER	AO	MARKS
2(a)	5. (1)	AO2	1
2(b)	10. (1)	AO2	1
2(c)	By value passes a copy, so the original is unaffected (1); by reference passes the original (its location), so changes persist after the subroutine ends (1). (2)	AO1	2
2(d)	By reference (1); by value is safer because the subroutine cannot accidentally change the caller's variable (1). (2)	AO2	2

Q	ANSWER	AO	MARKS
3(a)	Global: `score` (1); local: `bonus` (1). (2)	AO2	2
3(b)	The part of the program (1); in which a variable can be accessed/used (1). (2)	AO1	2
3(c)	When the procedure `addBonus` ends/returns. (1)	AO2	1

Q	ANSWER	AO	MARKS

4(a)	Indicative solution (award for logic):
	`function isEven(n)`
	`  if n MOD 2 == 0 then`
	`  return True`
	`  else`
	`  return False`
	`  endif`
	`endfunction`
	Uses `MOD 2` (or equivalent even test) (1); returns True when even (1); returns False otherwise (1). (Accept `return n MOD 2 == 0` for full marks.) (3)
		AO3	3

4(b)	A function, because it returns a value (True/False). (1)	AO1	1

Q	LEVELS-OF-RESPONSE MARK SCHEME (9 MARKS)	AO	MARKS
5	Mark using the levels descriptors below. AO1 (knowledge of scope/parameters), AO2 (application), AO3 (balanced comparison & conclusion). **LEVEL** / **DESCRIPTOR** Level 3 (7–9): Thorough comparison covering **both** local/parameters and globals, well applied to reliability/maintainability, with a **justified conclusion**. Accurate terminology. Level 2 (4–6): Both sides covered with some development; a basic conclusion. Level 1 (1–3): A few points, largely one-sided, little development. 0: Nothing creditworthy. **Indicative content** (credit any valid point): • Locals/parameters: self-contained subroutines; no accidental interference; no naming clashes; reusable/modular; easier to test & debug; memory freed when subroutine ends. • Globals: accessible everywhere, convenient for a value genuinely needed throughout (e.g. a game score); but any code can change them → harder to trace bugs in large programs; risk of clashes. • Reliability/maintainability: parameters make data flow explicit and testable; overusing globals makes programs fragile. • A justified conclusion (prefer locals/parameters; reserve globals for genuinely shared state).	AO1 ×3 AO2 ×3 AO3 ×3	9

Total for paper: 30 marks