2.3 ALGORITHMS · 2.3.1(F)

Linear & binary search — Mark scheme

30 marks · spec 2.3.1(f)

AO key: AO1 = knowledge · AO2 = application · AO3 = programming. Trace and complexity answers verified. Code is indicative.

Q	ANSWER	AO	MARKS
1(a)	Checks each item in turn from the start (1); until the target is found or the end is reached (1). (2)	AO1	2
1(b)	O(n). (1)	AO1	1
1(c)	It works on an unsorted list (no need to sort first) / simpler. (1)	AO1	1
1(d)	The list must already be sorted. (1)	AO1	1
1(e)	O(log n). (1)	AO1	1

Q	ANSWER	AO	MARKS
2(a)	<table><tr><th>LOW</th><th>HIGH</th><th>MID</th><th>LIST[MID]</th></tr><tr><td>0</td><td>8</td><td>4</td><td>34</td></tr><tr><td>0</td><td>3</td><td>1</td><td>11</td></tr><tr><td>2</td><td>3</td><td>2</td><td>19</td></tr><tr><td>3</td><td>3</td><td>3</td><td>23</td></tr></table> Row 1 mid=4, list[mid]=34 (1); 23<34 → high=3, row 2 (0,3,1,11) (1); 23>11 → low=2, row 3 (2,3,2,19) (1); 23>19 → low=3, row 4 (3,3,3,23) (1); correct mid values throughout (1); correct list[mid] values throughout (1). (6)	AO2	6
2(b)	Found at index 3 (1); in 4 steps (1). (2)	AO2	2

Q	ANSWER	AO	MARKS

3(a)	Indicative solution (award for logic):
	`found = False`
	`for i = 0 to length(list) - 1`
	`  if list[i] == target then`
	`  found = True`
	`  endif`
	`next i`
	`found` initialised to False (1); loop over all items (1); compare each item with `target` (1); set `found = True` when matched (1). (4)
		AO3	4

3(b)	n. (1)	AO2	1
3(c)	When the target is the last item, or is not in the list. (1)	AO2	1

Q	ANSWER	AO	MARKS
4(a)	Binary search (1); it is O(log n), far fewer steps on a large list, and the list is already sorted (1). (2)	AO2	2
4(b)	Linear search (1); the list is unsorted (binary search can't be used) and small, so sorting first would not be worth it (1). (2)	AO2	2

Q	ANSWER	AO	MARKS
5(a)	About 10 steps (1); because each step halves the list ($\log_2 1000 \approx 10$) (1). (2)	AO2	2
5(b)	It decides which half to discard by comparing with the middle item (1); on unsorted data the wrong half could be discarded, giving incorrect results (1). (2)	AO2	2
5(c)	Sort the list first (once) so binary search can be used repeatedly (1); drawback: sorting takes time/processing, and the list must be re-sorted if it changes (1). (2)	AO2	2

Total for paper: 30 marks