AQA Computer Science A-Level

4.4.4 Classification of algorithms

Past Paper Mark Schemes

Additional Specimen Paper 1

01	1	**Mark is for AO1 (knowledge)** **1 mark**: B;	1

01	2	**Mark is for AO1 (knowledge)** **1 mark**: A;	1

01	3	**Mark is for AO1 (understanding)** It demonstrates that there are some (well-defined) problems that cannot be solved by a computer;	1

02	4	**Mark is for AO2 (analyse)** nlog n **A.** O(nlog n) **A.** $O(n\log_2 n)$ **A.** $n\log_2 n$ **NE.** log(n)	1

Additional Specimen Paper 2

08	**1**	**Mark is for AO2 (apply)**	
		$\mathbb{N}$ //{ z | z ∈ $\mathbb{N}$ and z<y }; **A.** Natural numbers **A.** { z | z ∈ $\mathbb{N}$ and z>=0 and z<y }	1

June 2011 Comp 3

1	(c)		<table><tr><th>Order of complexity</th><th>Tick one box</th></tr><tr><td>$O(\log_2 n)$</td><td>✓</td></tr><tr><td>$O(n)$</td><td></td></tr><tr><td>$O(n^2)$</td><td></td></tr></table> ***Do not award mark if more than one box ticked***	1

June 2012 Comp 3

3	(a)		Space / Memory (complexity); **A** amount of memory used	1

3	(b)	(ii)	$O(n^2)$;	1
3	(b)	(iii)	**MARKS CAN ONLY BE AWARDED IF CORRECT ANSWER FOR PART 3 (b) (ii)** **Alternative 1:** Algorithm has nested loops // two loops with one inside the other; **A** reference to inner and outer loops Each loop repeats N times; **Alternative 2:** The (basic) operation / If statement / file read / comparison is carried out N^2 times; because it is inside nested loops // because each loop executes N times; **Alternative 3:** Each of the (N) entries is compared to each of the (N) others // each entry is compared N times; so N^2 (**A** N*N) comparisons/operations are required // $N*N=N^2$; **A** uppercase or lowercase n **A** answers where examples are used instead of N and N^2, e.g. 3 and 9. **A** check as alternative to comparison **MAX 2**	2

11	(a)		Is it possible in general to ***write a program/algorithm;*** that can tell, given any program and its inputs and without ***running/executing the program;***, whether the given program with its given inputs will halt? **A** "it" in second reference to program. **A** "create a Turing machine" for "write an algorithm"	2

11	(b)		Shows that some problems are non-computable / undecideable // shows that some problems cannot be solved by a computer/algorithm; In general, inspection alone cannot always determine whether any given algorithm will halt for its given inputs // a program cannot be written that can determine whether any given algorithm will halt for its given inputs; **A** it is not computable **MAX 1**	1

June 2017 Paper 1

03	**1**	**Mark is for AO1 (knowledge)** Merge sort;	**1**
03	**2**	**Mark is for AO1 (understanding)** 4;	**1**
03	**3**	**Mark is for AO1 (knowledge)** n^2 // O (n^2); **A.** other ways of indicating n^2 e.g. n^2 **A.** On^2	**1**

03	**4**	**Marks are for AO1 (understanding)** In each pass through the list n items will be examined; There will be (at most) n passes through the list;	**2**

05	4	**1 mark for AO1 (knowledge)** a heuristic approach employs a method of finding a solution that might not be the best; **1 mark for AO1 (understanding)** algorithm might need to consider visiting less/fewer cells/co-ordinates // algorithm might use knowledge of the domain to cut-down the search space // algorithm might consider visiting certain cells/coordinates first;	2

June 2013 Comp 3

6	(a)		Most efficient: C // O(n) **A.** n B // O(n^2) **A.** n^2 Least efficient: A // O(a^n) **A.** a^n	1	

6	(b)	(i)	The problem can be solved; But not in polynomial time // only in exponential (or worse) time // it takes an unreasonable amount of time to do so // can't be solved quickly enough for it to be useful; **A** takes too long for a computer to solve but **NE** just takes a long time **A** "algorithm exists" for can be solved **A** answers relating to space rather than time **TO** of the solving mark, if states that can be solved in polynomial/reasonable time	2	

6	(b)	(ii)	**Problem** / **Intractable? (Tick One)** The travelling salesman problem. ✓; The problem of sorting a list of names into alphabetic order. The Halting problem. **A** alternative indicators for ticks **Do not award mark if more than one box is ticked.**	1	

Specimen Paper 1

03	6	**All marks AO1 (knowledge)** **1 mark (1 from):** The problem can be solved // algorithm exists for problem; But it cannot be solved in polynomial time // but not quickly enough to be useful; **Max 2** **1 mark:** It takes an unreasonable amount of time; to solve; **A** Too long time but **R** Long time	2

03	7	**All marks AO1 (understanding)** **1 mark:** Use of heuristic; algorithm that makes a guess based on experience; That provides a close-to-optimal solution/approximation; that only works in some cases; **A** non-optimal Example of heuristic method eg hill-climbing/stochastic/local improvement/greedy algorithms/simulated annealing/trial and error/any reasonable example; **1 mark:** Relax some of the constraints on the solution; **A** Solve simpler version of problem	2

04	4	**Mark is for AO1 (understanding)**	1

		`O(k^n);` **A** `k^n`	

04	5	**Mark is for AO1 (knowledge)** `O(log n);` **A** `log n`	1

04	6	**Mark is for AO1 (knowledge)** `O(1);` **A** `1`	1

04	7	**Mark is for AO1 (knowledge)** `O(n);` **A** `n`	1

04	8	**All marks AO1 (understanding)** **1 mark:** As the size of the list increases the time taken to search for an item increases; at the same rate; // **1 mark:** A linear search looks at each item in the list in turn (until it reaches the end of the list or the item being searched for is found); so if there are n items in the list the worst case would be n comparisons;	2