AQA Computer Science A-Level

4.4.2 Regular languages

Past Paper Mark Scheme

Additional Spec Qs Paper 1

04	1	**All marks AO1 (knowledge)** A set is an unordered collection (of values); in which each value occurs at most once // with no duplicates;	2

04	2	**Mark is for AO2 (apply)** S3 // S5;	1

04	3	**Mark is for AO2 (apply)** S5;	1

04	4	**Mark is for AO2 (apply)** The difference between set S5 and S1 // the difference between set S2 and S1 // S5 – S1 // S2 – S1;;	1

04	5	**All marks AO2 (apply)** {(a, c); (b, c);} // {(c, a); (b, c);} // {(x, y) : x ←S1, y←S6};; ,	2

		MAX 1 if additional pairs listed	

04	6	**All marks AO2 (apply)** a | ab; ab?; **A.** any correct answer	2

June 2011 Comp 3

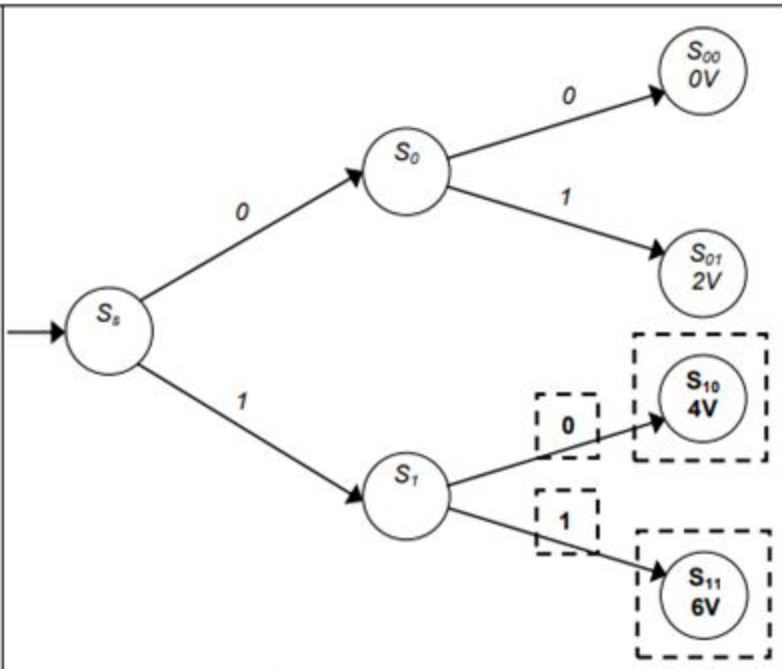
4	(a)		<table><tr><td>**Current State**</td><td>S_1</td><td>S_1</td><td>S_2</td><td>S_2</td><td>S_3</td><td>S_3</td><td>S_4</td><td>S_4</td><td>S_5</td><td>S_5</td></tr><tr><td>**Input Symbol**</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>***0***</td><td>***1***</td><td>***0***</td><td>***1***</td></tr><tr><td>**Next State**</td><td>S_2</td><td>S_3</td><td>S_2</td><td>S_4</td><td>S_3</td><td>S_3</td><td>S_4</td><td>S_5</td><td>S_5</td><td>S_4</td></tr></table> **1 mark** for all four bolded columns correct **A** the two columns for S_4 either way round and similar for S_5	1
4	(b)		Accept/Accepting/Accepted (state) // Input (string) is accepted **A** if the FSA finishes in this state output is Yes **R** Stop state	1

4	(c)		<table><tr><th>Input String</th><th>String Accepted? (Yes/No)</th></tr><tr><td>101</td><td>***No***</td></tr><tr><td>000</td><td>***No***</td></tr><tr><td>010001101</td><td>***No***</td></tr><tr><td>0100011011</td><td>***Yes***</td></tr></table> **1 mark** for any two correct answers **2 marks** for all four answers correct	2
4	(d)		Strings that start with a 0; **A** does not start with 1 **R** starts with 00, 01, any statement of a specific second digit being required Followed by any sequence containing an odd number of 1s and zero or more 0s; **A** String with an odd number of 1s in it. **A** Numbers or bit patterns in place of 0s and 1s.	2

9	(a)	(i)	One or more a's followed by (a/one) b; **A** answers by example but must be at least ab, aab, aaab and show that the sequence continues.	1
9	(a)	(ii)	The strings ab or b // zero or one a's followed by (a/one) b	1
9	(a)	(iii)	A sequence of 0 or more occurrences of ab. **A** answers by example but must be at least the empty string, ab, abab, abababab and show that the sequence continues.	1

9	(b)	(i)	Clai?re // Clare|Claire **A** other valid possibilities e.g. Cla(ir|r)e, Cl(air|ar)e **A** use of different types of brackets	**1**
9	(b)	(ii)	10(0|1)*01 **1 mark** for the 10 at the start and 01 at the end **1 mark** for (0|1)* in the middle to produce a correct expression **A** use of different types of brackets Award **2 marks** for any other expression that would work	**2**

June 2012 Comp 3

4	(d)		 **1 mark** for labelling a transition arrow with 0 **1 mark** for labelling a transition arrow with 1 **1 mark** for labelling a state with the value 4V and a unique state name **1 mark** for labelling a state with the value 6V and a unique state name **MAX 2 if the states and transition arrow labels do not correspond** Note that: • The state names do not have to match those given here. • The voltage values can be followed by a V, the word Volts or nothing. • The zero and one on the transition arrows to the right of S_1 can be either way around e.g. 1 above 0 is okay.	4

12	(a)		ab^{+}c // abb*c // ab*bc; **I** ^ at start, $ at end of expression	1

12	(b)		(0|1)1* // (1|0)1* // [01]1*// [10]1* // [0|1]1* // [1|0]1* // 0 | (0?1^{+}) **I** ^ at start, $ at end of expression	1

June 2016 AS Paper 1

02	**1**	**All marks AO2 (apply)** **Mark as follows:** **1 mark:** one row correct **2 marks:** all rows correct	2

Input string	Accepted by FSM?
111011x	NO
1110x	YES
111001x	NO

02	**2**	**All marks AO2 (apply)** Strings that start with zero or more 1s; **A.** starts with any number of 1s as BOD which may or may not be followed by a 0; **A.** there can be at most one 0 in the string and end with an x; **A.** 'end' being by implication **NOTE:** 'ending with either x or 0x' **is worth two marks** **NOTE: MAX 2** if answer is not fully correct	3

June 2017 AS Paper 1

Qu			Marks
01	1	**All marks AO2 (apply)** **1 mark** per two correct labels (round down). **I.** case **Note**: each label must only be used once (if given more than once, reject all occurrences).	4

Event	Label(s)
Correct code keyed	F
Door pulled open	B
Door pushed shut	A
New code keyed	E
Press C	d, g (**I.** order)
Press E	h, c (**I.** order)

June 2017 Paper 1

02	1	**Mark is for AO2 (analyse)** Input string is a (valid) postcode followed by additional characters // the input string is not a valid (UK) postcode // the mail will not be put in any of the three vans; **NE.** the input string is not a valid IP postcode **A.** Postcode has additional characters at the end **A.** Postcode is too long	1
02	2	**Mark is for AO2 (analyse)** (The string represents) an IP postcode that is not for a location in the town of Ipswich // (The string represents) an IP postcode that is for a location near Ipswich // (The string represents) a postcode for a letter that needs to go in Van B; **NE.** valid postcode	1

02	3	**Mark is for AO2 (analyse)**
		
		(IP / two letters) followed by number, letter, (number, letter, letter) //
		(IP / two letters) followed by number between 5 and 9, number, (number, letter, letter) //
		IP followed by 0;
		
		A. postcodes that only have one letter at the start
			1

02	4	**Marks are for AO2 (apply)** `\a?\a;\d;(\a|\d)?;\d\a\a;` // `\a\a?;\d;(\a|\d)?;\d\a\a;` // `\a?\a;\d;(\d|\a)?;\d\a\a;` // `\a\a?;\d;(\d|\a)?;\d\a\a;` **Mark as follows:** **1 mark:** 1. regular expression can start with either one or two letters **R.** if more than two letters allowed **1 mark:** 2. regular expression has a numeric digit after the initial letters **A.** if more than the correct number of letters allowed // regular expression has a numeric digit before it allows a single, optional letter or numeric digit **1 mark:** 3. regular expression allows a single, optional letter or numeric digit after the first numeric digit in the expression // regular expression allows a single, optional letter or numeric digit before the numeric digit followed by exactly two letters at the end of the expression **1 mark:** 4. regular expression ends with a numeric digit followed by exactly two letters **MAX 3** if final answer is not correct **R.** any mark points after 2nd use of | metacharacter **A.** suitable alternatives to `\a` and `\d` e.g. use of `[A-Z]`, `[a-z]` or `[A-Za-z]` instead of `\a` and `[0-9]` instead of `\d` **DPT.** / instead of \	4

June 2012 Comp 1

09	<table><tr><th>Original State</th><th>Input</th><th>New State</th></tr><tr><td>S0</td><td>10</td><td>S10</td></tr><tr><td>S0</td><td>20</td><td>S20</td></tr><tr><td>S0</td><td>50</td><td>S50</td></tr><tr><td>S0</td><td>R</td><td>S0</td></tr></table> ; ; ; *Note: order of completed rows not important*	**3**

10	20, 20, 10; R, R, 50;	

	10, 20, 20; 20, 50, 50; 20, R, 50;	**MAX 4**

June 2013 Comp 1

06	11101110; **R.** 01110111	**1**

07	11101011; **DPT A.** 11010111	**1**

08	Get the two's complement (of a positive binary value) // Converts a positive binary value into its negative equivalent; **A.** It inverts all bits after the first 1 is received;	**1**

09	<table><tr><th>Input</th><th>Original State</th><th>Output</th><th>New State</th></tr><tr><td>0</td><td>S0</td><td>0</td><td>S0</td></tr><tr><td>1</td><td>S0</td><td>1</td><td>S1</td></tr><tr><td>0</td><td>S1</td><td>1</td><td>S1</td></tr><tr><td>1</td><td>S1</td><td>0</td><td>S1</td></tr></table> **Mark as follows:** S0 as original state for 2nd row; 1 as output for 3rd row; Final row correct;	**3**

7	(a)	(i)	❷ S_1 **A.** 1, State 1 ❸ S_T **A.** T, State T Both answers correct to get mark;	**1**

7	(a)	(ii)	$\delta (S_B, 0) = (S_0, x, \rightarrow)$; **A.** 0, x, → or 0 | x | → **R** if additional rules listed **I** minor transcription errors e.g. missing , (δ	**1**

Specimen AS Paper 1

01	**6**	**All marks AO2 (apply)** **1 mark:** Two rows of table completed correctly; **OR** **2 marks:** All three rows of table completed correctly; **A.** Alternative indicators for YES and NO	2

Input string	Accepted by FSM?
aaab	YES
abbab	NO
bbbbba	YES

01	**7**	**All marks AO2 (apply)** **1 mark:** a string containing zero or more (**A.** 'any number of') b characters; **1 mark:** and an odd amount of a characters; **N.E.** all strings containing an odd number of characters	2

Specimen Paper 1

02	1	**Mark is for AO1 (understanding)** **1 mark:** Table completed as above **I.** order of rows	1

Original state	Input	New state
S3	0	S4
S3	1	S2

02	2	**All marks AO2 (analyse)** (0|1)*((00)|(11))(0|1)* **Mark as follows:** **1 mark:** (0|1)* at start; **1 mark:** (00)|(11); **1 mark:** (0|1)* at end; **Or** **Alternative answer** (0|1)*(11(0|1)*)|(00(0|1)*) **Mark as follows:** **1 mark:** (0|1)* at start; **1 mark:** (11(0|1)*); **1 mark:** |(00(0|1)*) at end;	3

		Maximum 2 marks: If final answer not correct. **A** any regular expression that correctly defines the language.	