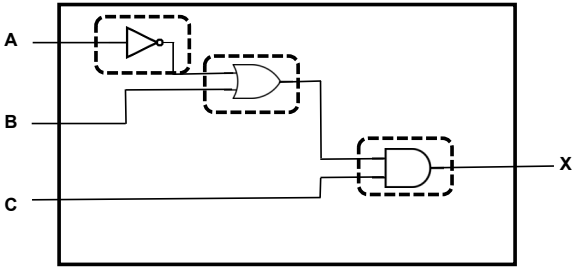
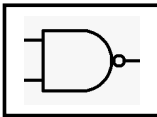


4(a)	One mark per bullet point - input of adult and child number - calculation of total without discount - check if total people are 6 or more ... apply discount to total - output of the total Example: <pre> OUTPUT "Please enter the number of adults " INPUT Adult OUTPUT "Please enter the number of children " INPUT Child Total ← (Adult * 9.99) + (Child * 6.99) IF (Adult + Child) >= 6 THEN Total ← Total - (Total * 15/100) ENDIF OUTPUT "The total cost is ", Total </pre>
4(b)	One mark per bullet point - Correct use of <code>ROUND</code> ... to 2 decimal places Example <pre>Total ← ROUND(Total, 2)</pre>
4(c)(i)	One mark per bullet point (NOT A OR B) - AND C (X =) (NOT A OR B) AND C
4(c)(ii)	 One mark per correct gate

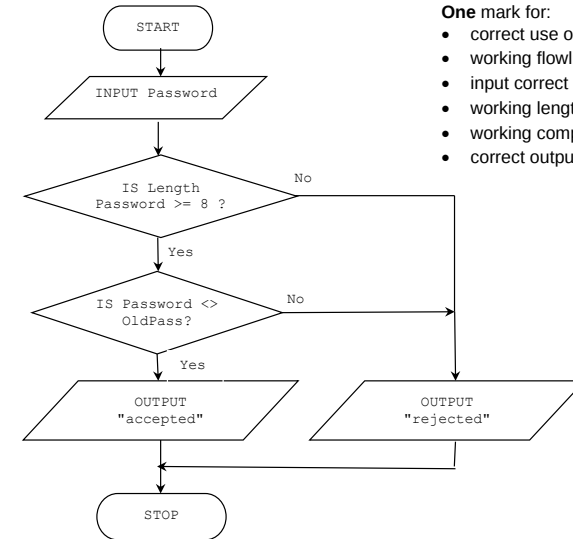
4(d)	- One mark for correct symbol - One mark for correctly completed table  <table border="1"> <thead> <tr> <th>A</th><th>B</th><th>X</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>1</td></tr> <tr> <td>0</td><td>1</td><td>1</td></tr> <tr> <td>1</td><td>0</td><td>1</td></tr> <tr> <td>1</td><td>1</td><td>0</td></tr> </tbody> </table>	A	B	X	0	0	1	0	1	1	1	0	1	1	1	0
A	B	X														
0	0	1														
0	1	1														
1	0	1														
1	1	0														

IGCSE Computer Science: w25 22

8	Use the tables for AO2 and AO3 below to award a mark in a suitable band using a best fit approach, then add up the total. Marks are available for: - AO2 (maximum 9 marks) - AO3 (maximum 6 marks) Data Structures required with names as given in the scenario: Arrays or lists <u>RandomNumber[]</u>, <u>CountedNumber[]</u> Requirements (techniques) R1 arrays and variables declared, then 100 000 random integers between 1 and 10, inclusive, generated and stored in an array, (array and variable declarations, iteration, random number generation, rounding and storage). R2 frequency of each random integer counted and results stored in array, then results sorted into descending order (initialisation, selection, counting, sort, nested iteration, storage). R3 calculation of the chance of each random integer occurring rounded to 4 d.p. and results output (iteration, rounding and output). Example 15-mark answer in pseudocode <pre> // array and variable declaration DECLARE RandomNumber : ARRAY[1:100000] OF INTEGER DECLARE CountedNumber : ARRAY[1:10, 1:2] OF INTEGER DECLARE Index1 : INTEGER DECLARE Index2 : INTEGER DECLARE Swap : BOOLEAN DECLARE Temp1 : INTEGER DECLARE Temp2 : INTEGER // generate 100000 random integers between 1 and 10, inclusive, // and store them in an array FOR Index1 ← 1 TO 100000 RandomNumber[Index1] ← ROUND(RANDOM() * 9, 0) + 1 NEXT Index1 </pre>
---	--

8	<pre>// initialising the CountedNumber array FOR Index1 ← 1 TO 10 CountedNumber[Index1, 1] ← Index1 CountedNumber[Index1, 2] ← 0 NEXT Index1 // counting the frequency of each of the random integers // and storing the frequencies in an array FOR Index1 ← 1 TO 10 FOR Index2 ← 1 TO 100000 IF RandomNumber[Index2] = CountedNumber[Index1, 1] THEN CountedNumber[Index1, 2] ← CountedNumber[Index1, 2] + 1 ENDIF NEXT Index2 NEXT Index1 // performing sort on frequencies, so the possible random numbers // are stored in descending order of frequency Swap ← TRUE WHILE Swap DO // sort is terminated when no swaps take place during a pass Swap ← FALSE FOR Index1 ← 1 TO 9 IF CountedNumber[Index1, 2] < CountedNumber[Index1 + 1, 2] THEN Temp1 ← CountedNumber[Index1, 1] Temp2 ← CountedNumber[Index1, 2] CountedNumber[Index1, 1] ← CountedNumber[Index1 + 1, 1] CountedNumber[Index1, 2] ← CountedNumber[Index1 + 1, 2] CountedNumber[Index1 + 1, 1] ← Temp1 CountedNumber[Index1 + 1, 2] ← Temp2 Swap ← TRUE ENDIF NEXT Index1 ENDWHILE</pre>
8	<pre>// outputting the results, with chances for each random number // calculated to 4 d.p. // results in descending order of frequency FOR Index1 ← 1 TO 10 OUTPUT "The chance of ", CountedNumber[Index1, 1], " occurring is: " OUTPUT ROUND(CountedNumber[Index1, 2]/100000, 4) NEXT Index1</pre> Alternative answer for first 20 lines after declarations, where RandomNumber[] is populated and frequency counted in the same loop. <pre>// initialising the CountedNumber array FOR Index1 ← 1 TO 10 CountedNumber[Index1, 1] ← Index1 CountedNumber[Index1, 2] ← 0 NEXT Index1 // generate random integers in an array, // counting and storing the frequencies in an array FOR Index1 ← 1 TO 100000 Temp1 ← ROUND(RANDOM() * 9, 0) + 1 RandomNumber[Index1] ← Temp1 CountedNumber[Temp1, 2] ← CountedNumber[Temp1, 2] + 1 NEXT Index1 // performing sort on frequencies, so the possible random numbers</pre>

8(a)



- One mark for:**
- correct use of flowchart symbols
 - working flowlines
 - input correct
 - working length check
 - working comparison
 - correct output messages

8(b)

- One mark for each point**
- OPENFILE MyPassword.txt FOR WRITE
 - WRITEFILE MyPassword.txt, Password
 - CLOSEFILE MyPassword.txt

8(c)

- One mark for each point**
- needs to be retrieved on demand // saved for a later date
 - storage must be non-volatile