You will notice in the Boolean algebra, three new symbols; these have the following meaning:

- . represents the AND operation
- + represents the OR operation
- a bar (above the letter or letters, e.g. $\overline{a}$) represents the NOT operation.

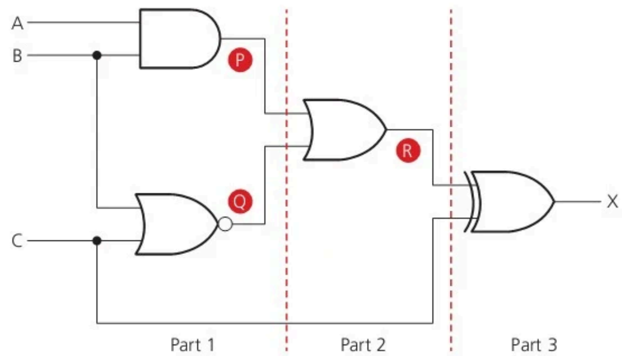
10.3 Logic circuits, logic expressions, truth tables and problem statements

When logic gates are combined together to carry out a particular function, such as controlling a robot, they form a logic circuit. The following eight examples show how to carry out the following tasks:

- Create a logic circuit from a:
  - problem statement (examples 6 and 7)
  - logic or Boolean expression (examples 3 and 8)
  - truth table (examples 4 and 5)
- Complete a truth table from a:
  - problem statement (examples 6 and 7)
  - logic or Boolean expression (examples 3 and 8)
  - logic circuit (example 1)
- Write a logic or Boolean expression from a:
  - problem statement (examples 6 and 7)
  - logic circuit (example 2)
  - truth table (examples 4 and 5).

Example 1

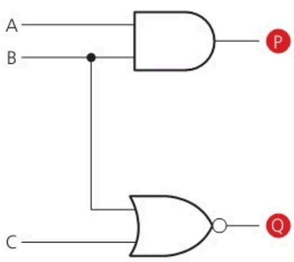
Produce a truth table for the following logic circuit (note the use of black circles at the junctions between wires):



▲ **Figure 10.8**

There are three inputs to this logic circuit, therefore, there will be eight possible binary values that can be input.

To show stepwise how the truth table is produced, the logic circuit has been split up into three parts, as shown by the dotted lines, and intermediate values are shown as P, Q and R.



▲ **Figure 10.9**

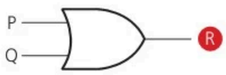
Part 1

This is the first part of the logic circuit; the first task is to find the intermediate values P and Q.

The value of P is found from the AND gate where the inputs are A and B. The value of Q is found from the NOR gate where the inputs are B and C. An intermediate truth table is produced using the logic function descriptions in Section 10.2.

▼ **Table 10.8**

input values			Output values	
A	**B**	**C**	**P**	**Q**
0	0	0	**0**	**1**
0	0	1	**0**	**0**
0	1	0	**0**	**0**
0	1	1	**0**	**0**
1	0	0	**0**	**1**
1	0	1	**0**	**0**
1	1	0	**1**	**0**
1	1	1	**1**	**0**



▲ **Figure 10.10**

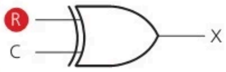
Part 2

The second part of the logic circuit has P and Q as inputs and the intermediate output, R:

This produces the following intermediate truth table. (Note: even though there are only two inputs to the logic gate, we have generated eight binary values in part 1 and these must all be used in this second truth table).

▼ **Table 10.9**

Inputs		Output
P	**Q**	**R**
0	1	**1**
0	0	**0**
0	0	**0**
0	0	**0**
0	1	**1**
0	0	**0**
1	0	**1**
1	0	**1**



▲ **Figure 10.11**

Part 3

The final part of the logic circuit has R and C as inputs and the final output, X:

This gives the third intermediate truth table:

▼ **Table 10.10**

Inputs		Output
R	**C**	**X**
1	0	**1**
0	1	**1**
0	0	**0**
0	1	**1**
1	0	**1**
0	1	**1**
1	0	**1**
1	1	**0**

Putting all three intermediate truth tables together produces the final truth table, which represents the original logic circuit:

▼ **Table 10.11**

Input values			Intermediate values			Output
A	**B**	**C**	**P**	**Q**	**R**	**X**
0	0	0	*0*	*1*	*1*	1
0	0	1	*0*	*0*	*0*	1
0	1	0	*0*	*0*	*0*	0
0	1	1	*0*	*0*	*0*	1
1	0	0	*0*	*1*	*1*	1
1	0	1	*0*	*0*	*0*	1
1	1	0	*1*	*0*	*1*	1
1	1	1	*1*	*0*	*1*	0

The intermediate values can be left out of the final truth table, but it is good practice to leave them in until you become confident about producing the truth tables. The final truth table would then look like this:

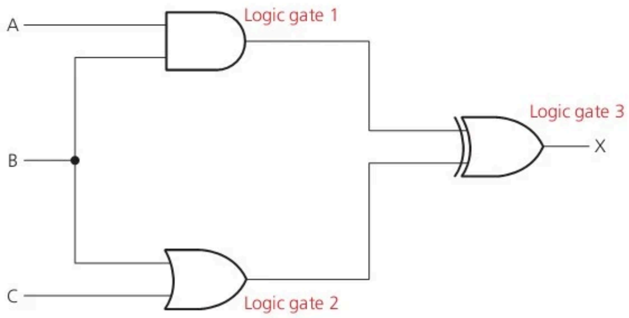
▼ **Table 10.12**

Input values			Output
A	**B**	**C**	**X**
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Example 2

Write logic expressions from the following logic circuits:

a



▲ **Figure 10.12**

The first action is to look at the gates connected to the inputs A, B and C:

logic gate 1: (A AND B)

logic gate 2: (B OR C)

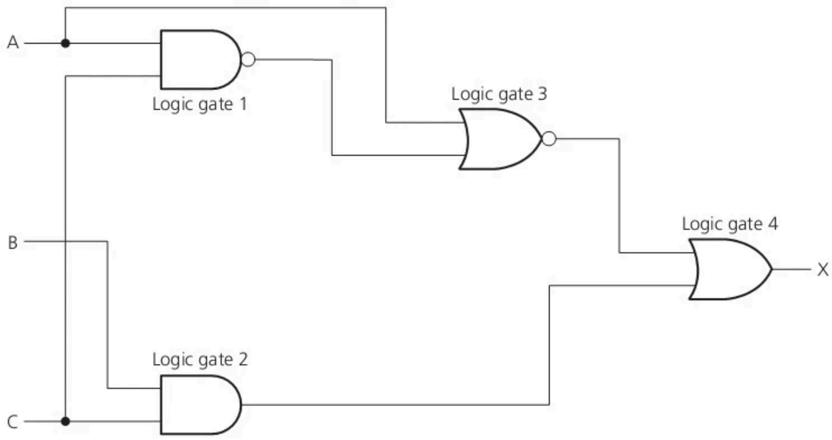
We then join these together using logic gate 3:

[(A AND B)] XOR [(B OR C)] which gives us the required logic expression.

(Note: the square brackets "[ ]" in the expression are not necessary and are used here just for clarity.)

This would be written as: (A AND B) XOR (B OR C)

b



▲ **Figure 10.13**

Again, we will do this in the order of logic gates 1 and 2 first (connected to the three inputs):

logic gate 1: (A NAND C)

logic gate 2: (B AND C)

However, logic gate 3 is also connected to one of the inputs so that should be done next:

logic gate 3: (logic gate 1) NOR A

If we replace (logic gate 1) by the logic expression above, we get:

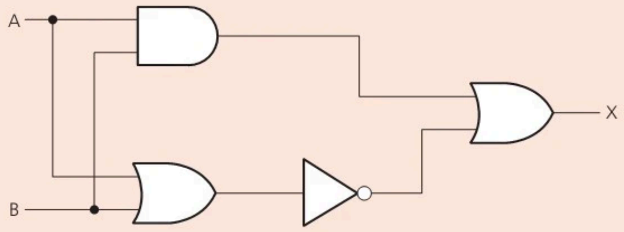
((A NAND C) NOR A)

Finally, we can join all these together using:

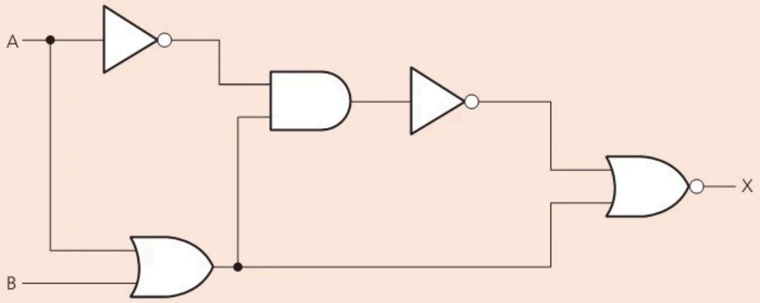
logic gate 4: ((A NAND C) NOR A) OR (B AND C)

Activity 10.2

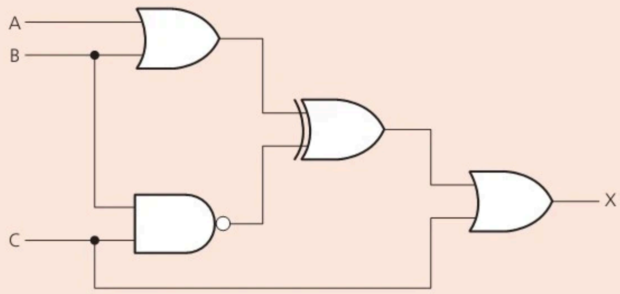
1 Produce truth tables from the following logic circuits:

a 

▲ Figure 10.14

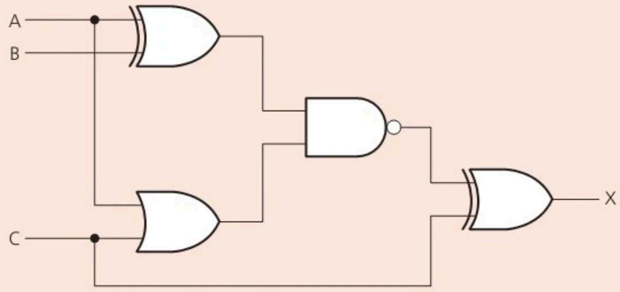
b 

▲ Figure 10.15

c 

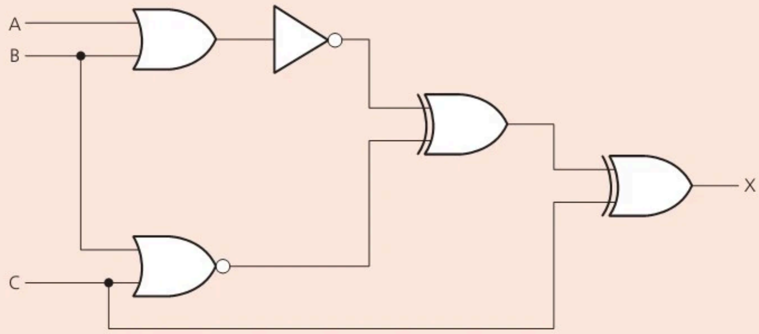
▲ Figure 10.16

d



▲ **Figure 10.17**

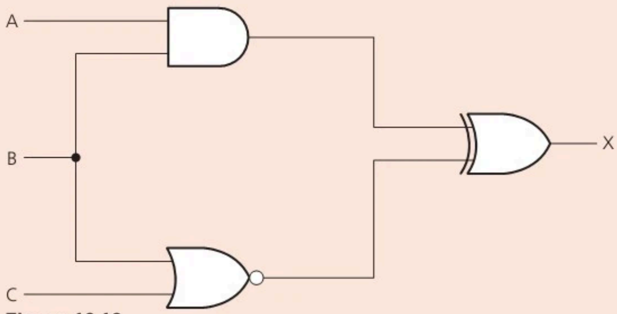
e



▲ **Figure 10.18**

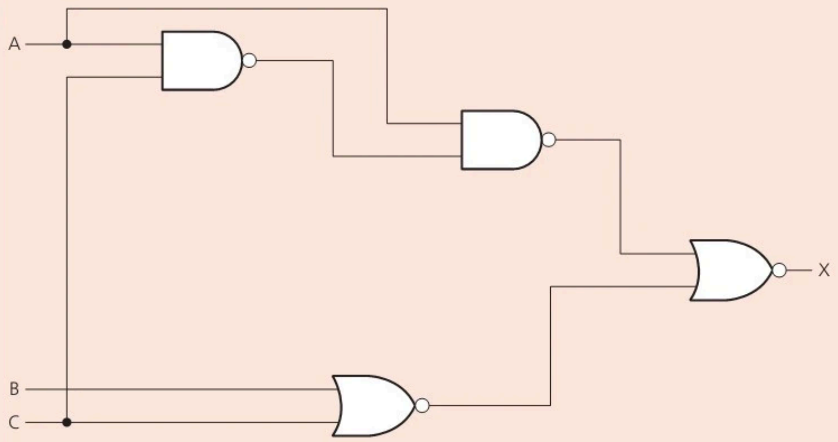
2 Write logic expressions for the following logic circuits:

a



▲ **Figure 10.19**

b



▲ Figure 10.20

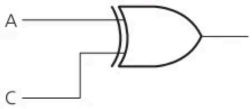
Example 3

A logic circuit can be represented by the following logic expression: (A XOR C) OR (NOT C NAND B)

Produce a logic circuit and a truth table from the above statement.

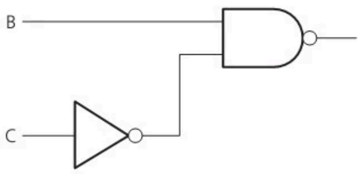
In this example we have a connecting logic gate which is OR.

So, if we produce one half of the circuit from (A XOR C) we get:



▲ Figure 10.21

The other half of the circuit is found from (NOT C NAND B):



▲ Figure 10.22