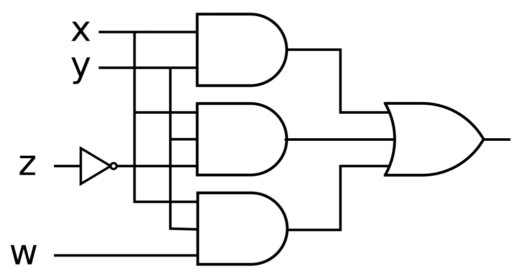
Note: This assignment is based on material covered in Brown and Vranesic 2nd Edition Sections 2.3 (truth tables), 2.4 (logic gates and networks), 2.5 (Boolean algebra), 2.6.1 (SoP and PoS forms; canonical forms), 2.7 (NAND and NOR networks), 2.8 (design examples).

Problem 1: The Boolean expression below is from Midterm Exam 1.

(*a*) Draw a logic diagram (show the gates) corresponding to the following Boolean expression. (As written, don't simplify it first.) Use AND, OR, and NOT gates.

$$x \cdot y + x \cdot y \cdot z' + y \cdot w \cdot x$$

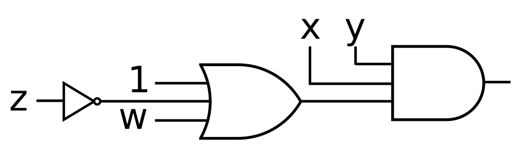
Solution appears below.



(*b*) In the expression below xy has been factored out. Draw a logic diagram corresponding to this expression. As far as the diagram is concerned, treat the 1 as an input (x, y, z, and w are the real inputs). *Note: If this is solved correctly then something in the diagram should look foolishly wasteful. See the next part.*

$$x \cdot y \cdot (1 + z' + w)$$

Solution appears below.



(*c*) The simplified form of the expression is just $x \cdot y$. Explain why that's much more obvious in the second diagram above (where xy was factored out) than in the first.

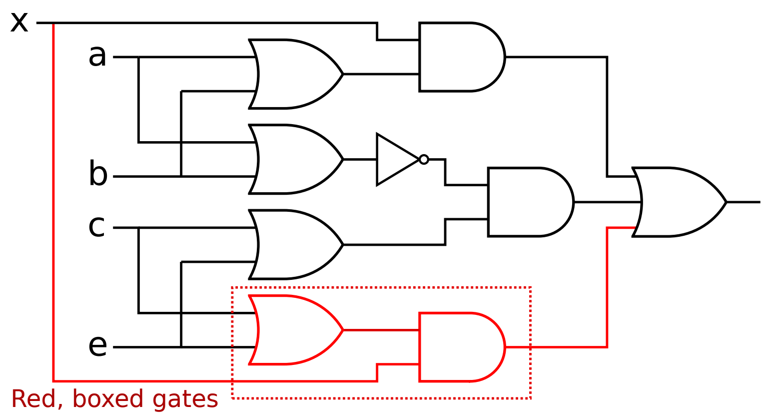
Because there's a 1 input to an OR gate and so the output of the OR gate will be the constant 1 which is also the input to the AND gate. The OR gate could be replaced with the constant 1 but even better the OR gate and the input to the AND gate can be eliminated, leaving just $x \cdot y$.

Problem 2: The expression below is also from Midterm Exam 1.

(a) Draw a logic diagram (show the gates) corresponding to the following Boolean expression. (As written, don't simplify it first.) Use AND, OR, and NOT gates.

$$x \cdot (a+b) + (a+b)' \cdot (c+e) + (c+e) \cdot x$$

The logic diagram appears below.



(b) The expression above can be simplified by applying the consensus theorem ($i \cdot y + i' \cdot z + y \cdot z = i \cdot y + i' \cdot z$). Simplify it (or just look at the test solution if you honestly don't need the practice). Indicate the change in the logic diagram from the previous part.

The consensus theorem (with $i \rightarrow (a+b)$) indicates that term $(c+e) \cdot x$ is unnecessary, yielding $x \cdot (a+b) + (a+b)' \cdot (c+e)$. The red and boxed parts of the diagram above are the gates corresponding to the eliminated term, $(c+e) \cdot x$.

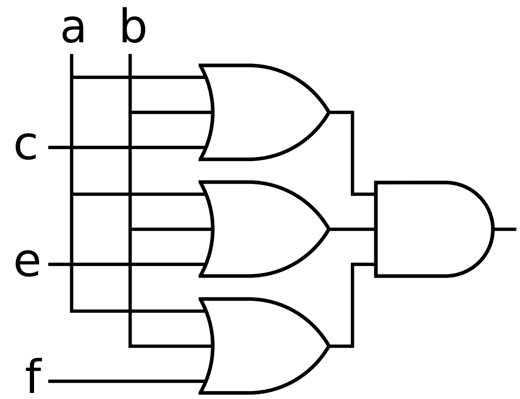
Note that the diagram could be simplified further by eliminating one of the OR gates realizing $a+b$ and then using the output of the other one in two places.

Problem 3: The expression below is also from Midterm Exam 1.

(a) Draw a logic diagram (show the gates) corresponding to the following Boolean expression. (As written, don't simplify it first.) Use AND, OR, and NOT gates.

$$(a + b + c) \cdot (a + b + e) \cdot (a + b + f)$$

The logic diagram:

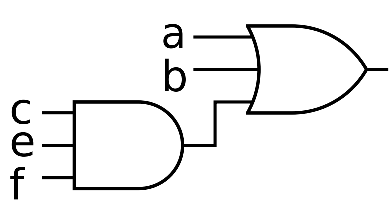


(b) Simplify the expression above by applying the rule $(x + i)(x + j) = x + ij$ several times. Note that x need not correspond to just one variable in the expression. As with the previous problem, look at the exam solution if you don't need the practice.

The simplified expression is $a + b + c \cdot e \cdot f$.

(c) Show a logic diagram for the simplified expression.

The logic diagram:



(d) Look back and forth at the two logic diagrams and think to yourself, "Both circuits do the same thing, but they look so different. Gee."

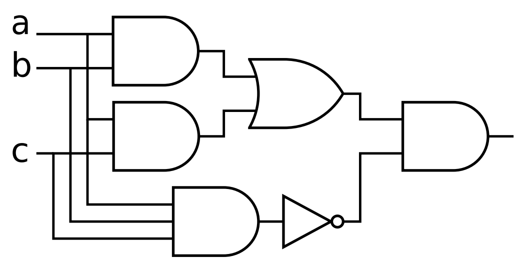
"Both circuits do the same thing, but they look so different. Gee."

Problem 4: Consider the expression below, which did not, incidentally, appear on the exam.

$$(a \cdot b + a \cdot c) \cdot (a \cdot b \cdot c)'$$

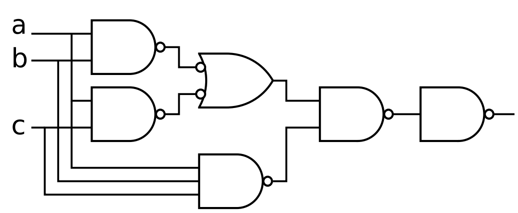
(*a*) Draw a logic diagram (show the gates) corresponding to the Boolean expression above. (As written, don't simplify it first.) Use AND, OR, and NOT gates.

The logic diagram:



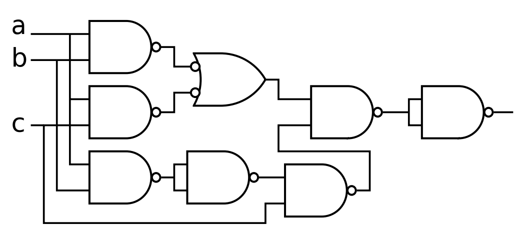
(*b*) Repeat the problem above, but use only NAND gates. The NAND gates can have any number of inputs.

The logic diagram appears below. Note that a one-input NAND gate is being used. That exactly the same as a NOT gate (except for the symbol we use and what we call it). Also notice that one of the NAND gates is drawn as an OR gate with bubbled inputs. That's okay because $(ab)' = a' + b'$.



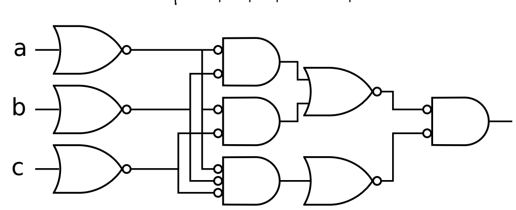
(*c*) Repeat the problem above, but use only two-input NAND gates.

The solution appears below. Notice that the single three-input NAND gate in the lower left of the diagram above has been replaced by three NAND gates, one of which serves as a NOT gate.



(*d*) Repeat the problem above, but use only NOR gates. The NOR gates can have any number of inputs.

The solution appears below. Notice that the inputs, a, b, and c, each have to be inverted.



Problem 5: The expression below is from the last problem, written with different notation.

$$(ab + ac)\overline{abc}$$

(a) Construct a truth table for this expression.

```
SOLUTION
a b c    val
------+-----
0 0 0 !  0
0 0 1 !  0
0 1 0 !  0
0 1 1 !  0
1 0 0 !  0
1 0 1 !  1
1 1 0 !  1
1 1 1 !  0
```

(b) Using the truth table, write the expression in minterm canonical form. (Show a Boolean expression don't use a shorthand such as $\sum_{x,y,z,w} m(12, 15)$.)

The Boolean expression in minterm canonical form is $a\overline{b}c + ab\overline{c}$.

(c) Using the truth table, write the expression in maxterm canonical form.

The Boolean expression in minterm canonical form is:

$$(a + b + c)(a + b + \overline{c})(a + \overline{b} + c)(a + \overline{b} + \overline{c})(\overline{a} + b + c)(\overline{a} + \overline{b} + \overline{c}).$$

Problem 6: A seven-segment display consists of rectangular lights, called *segments*, arranged to form an 8. If all segments are lit an 8 will appear. If all but the middle segment is lit a zero appears, etc. Let variables x_3, x_2, x_1, x_0 hold the four bits of a number represented in BCD (just one digit), with x_3 being the most significant bit.

(a) Construct a truth table for the Boolean function that indicates whether the topmost segment is lit. That is, the Boolean function should be 1 if $x_3 \cdots x_0 = 1001_2$ because for a nine the top segment is lit. The function should be 0 if $x_3 \cdots x_0 = 1_2$ because for the digit 1 the top segment is not lit. (Those are just two inputs, the function needs to be specified for the other eight digits.) Be creative about what to do when $x_3 \cdots x_0 > 1001_2$.

The solution appears below. The problem was clear on what to display for input values 0-9, but left open what to do about 10-15. If we were guaranteed never to have inputs 10 or higher we could use don't cares to minimize the logic, but that's a topic covered after this homework was assigned. Anyway, the problem said to be creative, meaning think of what could be displayed for these other values. One useful thing is a minus sign. Other possibilities include the letters a through f (though a is a challenge on a 7-segment display.) The solution here displays a minus sign and some other symbols. The truth table describes what is to be displayed for values 10 and higher.

x3-x0		Seg	
0	0000	1	
1	0001	0	
2	0010	1	
3	0011	1	
4	0100	0	
5	0101	1	
6	0110	1	(or 0, depending on taste)
7	0911	1	
8	1000	1	
9	1001	1	
10	1010	1	Top segment only
11	1011	0	Minus sign. (Middle segment only)
12	1100	0	Bottom segment only.
13	1101	1	All three horizontal segments.
14	1110	0	Lower right segment.
15	1111	0	All segments off

(b) Based on the truth table write the Boolean function in either minterm canonical form or maxterm canonical form, whichever is shorter.

Maxterm canonical form is shorter (since six rows have zeros, which is less than half).

$$(x_3+x_2+x_1+\overline{x_0})(x_3+\overline{x_2}+x_1+x_0)(\overline{x_3}+x_2+\overline{x_1}+\overline{x_0})(\overline{x_3}+\overline{x_2}+x_1+x_0)(\overline{x_3}+\overline{x_2}+\overline{x_1}+x_0)(\overline{x_3}+\overline{x_2}+\overline{x_1}+\overline{x_0}) \blacksquare$$

.