Name: ______________________________

CS 33003 **Exam #1** **CompOrg**

Monday 21 September 1998

---

1. For each of the following statements, write the word "true" below the statement if it is true, and "false" below the statement if it is false. (5 points each = 20 points)

a. Mechanical adding machines usually have more significant digits than slide rules.

True

b. Mechanical adding machines can use logarithms to multiply and divide.

False

c. Early computers were used to compute artillery tables during World War II.

True

d. Before becoming president of IBM, Thomas Watson was a highly successful stockbroker.

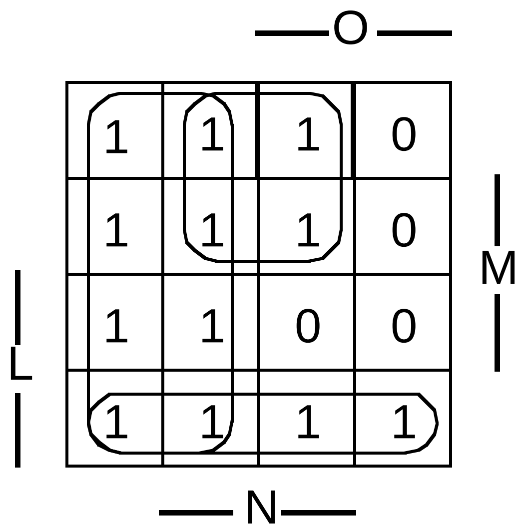
False

2. Give the truth table for a combinational circuit that takes a two-bit input labeled A_1A_0 and a two-bit input labeled B_1B_0 and produces a one-bit output C that is true (1) if the value A_1A_0 is numerically *greater than* the value B_1B_0 and is false (0) otherwise. (15 points)

A1	A0	B1	B0	C
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

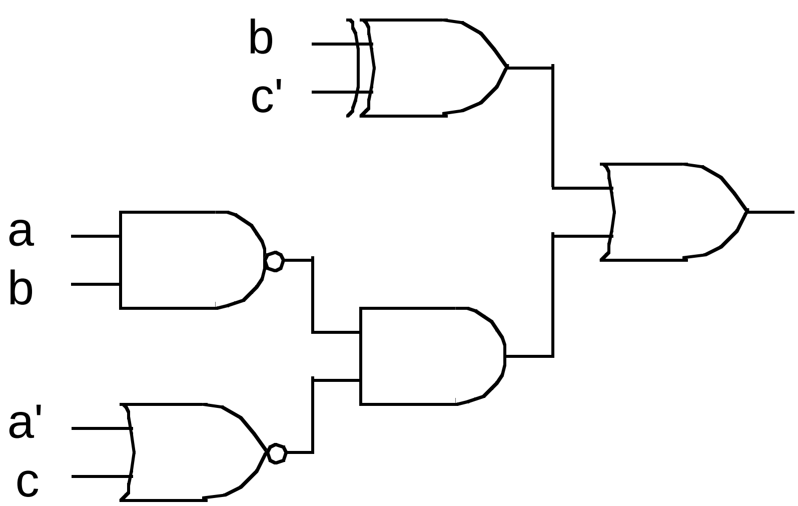
Name: ______________________________

3. Given the Karnaugh Map below, draw the appropriate ovals on the map, and write the simplified two-level sum-of-products expression to the right of the map. (20 points)



output = O' + L'N + LM'

4. Draw the combinational circuit that *directly* corresponds to the Boolean equation z = (b ⊕ c') + (ab)'(a'+c)' in the space below. (20 points)



Name: ________________________________

5. Perform the following conversions, showing your work. (5 points each = 15 points)

a. 29_{10} to base 2

29/2 = 14, rem 1
14/2 = 7, rem 0
7/2 = 3, rem 1
3/2 = 1, rem 1
1/2 = 0, rem 1 — the answer is 11101_2

b. 1100010101_2 to base 8

1100010101
= 001 100 010 101 (adding two leading zeros)
with 3 digits per octal digit, — the answer is 1425_8

c. $15A_{16}$ to base 10

(note that A = 10)
$1*16^2 + 5*16 + 10$
= 256 + 80 + 10 — the answer is 346_{10}

6. This question explores the difference between different types of encoding. (5 points each = 10 points)

a. Does it require more or less bits to store "23" as a ASCII character string than as a number? Explain your answer.

It would probably require more bits. Storing "23" as an ASCII string would require 7 bits for each character, for a total of 14 bits (or 16 bits, if each character is stored in a separate byte). Storing "23" as a number would require at least 5 bits ("23" is "10111" in binary) (or 8 bits if it is stored in a single byte). Either way, the numerical representation is smaller.

b. What is the basic idea behind Huffman encoding? (Note — I'm not asking for the entire algorithm or a detailed example, just the basic idea.)

Analyze the frequency with which the characters to be encoded occur, and use less bits than average for frequently-occurring characters, and more bits than average for seldom-occurring characters.