CSCI 1200, Spring 2003
Problem Set 1 (Non-programming portion)

Note: This portion of the problem set should be handed in **as hard-copy** to Elizabeth White's office. Please make your answers neat and readable (and don't forget to put your name on the pages!).

Problem 1.1 Building a function in digital logic

In class we used digital logic gates to build a binary adder, though we didn't quite complete the job: we didn't create the logic for the "carry-out bit". Here is the function that we need to write, where OUT is the output and A, B, and C are the input values:

OUT = (AB(NOT C)) OR (A(NOT B)C) OR ((NOT A)B C) OR (ABC)

In other words, the OUT value should be high if any two or all three of the input bits are high.

Write out the function using combinational logic gates (you can use AND and OR gates with more than two inputs if you wish).

Problem 1.2 A finite-state machine

The following table describes a four-state finite state machine. The four states are labeled START, A, B, and YES! Each state has two arrows (labeled 0 and 1) leading out of it and leading to another (or possibly the original) state. Here is the table—you might find it helpful to draw this out as a graph:

State:	On 0, go to:	On 1, go to:
START	YES!	State A
State A	State B	State A
State B	YES!	State A
YES!	YES!	State A

a. Starting with the START state, where does this finite-state machine end up when the input is:

000
100
1010
1001
1100
01010
11000

b. Think of these inputs as binary integers. How would you describe the numbers that this machine accepts (i.e., the numbers that end up in the YES! state)?

c. Create a three-state finite state machine, with states START, A, and YES! Your machine should take any sequence of 0s and 1s as inputs, but should only accept those sequences that have a "10" somewhere inside them. In other words, these sequences are accepted:

01000
00010
10
111011
0101011

But these are not:

00000
00111
1111
0
1