Solutions to Assignment 1

Question 1

decimal	binary	hex
8	1000	0x8
7	0111	0x7
16	10000	0x10
15	1111	0xf
256	100000000	0x100
255	11111111	0xff
237	1110 1101	0xEd

Question 2

binary	hex	decimal
1011	0x0b	11
1011 1011	0xbb	187
1000 0000	0x80	128
11 1100	0x3c	60
0011 1100	0x3c	60

Question 3

hex	binary	decimal
0x0e	0000 1110	15
0xe	0000 1110	15
0xAA	1010 1010	170
0xFA	1111 1010	250
0x40	0100 0000	64
0x18	0001 1000	24

Question 4

1. A bitwise 'and' operation with a '1' bit retains the value of that bit. A bitwise 'and' operation with a '0' bit always sets that bit to '0'.

```
  ( 0xaa & 0x0f )
= 0x0a
```

2.
```
  ( 0x3c & 0xf0 ) | ( 0x3c & 0x0f )
= ( 0x30 ) | ( 0x0c )
= 0x3c
```

3. Note that the && is the *logical* and operator.

```
  3 * ( 0xf0 && 0x0f )
= 3 * ( 0x1 )
= 0x3
```

4. An exclusive-or with a '1' bit inverts that bit.

```
  ( 0x3c ^ 0xff ) + ( 1 < 3 )
= ( 0xc3 ) + ( 1 )
= 0xc4
```

5.
```
  ~ ( 128 | ' ' )
= ~ ( 0x80 | 0x20 )
= ~ ( 0xa0 )
= 0x5f
```

6. Note that the || is the *logical* 'or' operator.

```
  128 || ( ' ' == 0x20 )
= 128 || ( 0x20 == 0x20 )
= 128 || 1
= 0x01
```

Question 5

```
/* Print the binary value of an integer
   less than 32768. We start at the
   largest applicable power of 2 and
   work our way down to the smallest
   power of 2.  For each power, if that
   power is "contained" in the number we
   remove it and print a '1', otherwise
   we print a '0'.  When all powers have
   been tested the result is that we
   have printed the binary
   representation of the number. */

#include <stdio.h>
```

```
void printbin ( int n )
{
  int p ;
  p = 16384 ;
  while ( p >= 1 ) {
    if ( n >= p ) {
      n = n - p ;
      printf ( "1" ) ;
    } else {
      printf ( "0" ) ;
    }
    p = p / 2 ;
  }
  printf ( "\n" ) ;
}
```

Question 6

This solution uses an index variable `n` which is also used to count the number of characters preceeding the terminating zero (null) character.

```
int len ( char s[] )
{
  int n ;
  n = 0 ;
  while ( s[n] != 0 ) {
    n = n + 1 ;
  }
  return n ;
}
```

Question 7

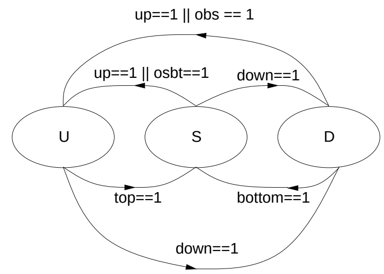
Using the labels given in the diagram the inputs are: `obs` (obstruction detected), `up`, `down`,`top`, and `bottom`. The outputs are: `raise`, and `lower`.

There are 3 combinations of outputs: (1) the "raise" motor turned on, (2) the "lower" motor turned on, and (3) neither motor turned on. The controller must therefore have at least 3 states. We will assign these states the labels `U`, `D`, and `S`, respectively. We will attempt to produce a correct design with this many states and add additional states if required to meet the specifications. The outputs for each state are thus:

state	`raise`	`lower`
U	1	0
D	0	1
S	0	0

Comparing the outputs required for each state and the state encoding, we see that both tables are the same so the output equations are: $U = A$ and $D = B$.

A simple state transition diagram would be as follows:



The diagram above (like the question itself) is ambiguous because the logical expressions for various state transitions are not mutually exclusive. For example, it is not clear what should happen if both the up and down inputs are '1'. These ambiguities need to be resolved before a controller can be implemented. The following state transition table unambiguously gives the transitions which are taken for each input condition. The symbol 'X' is used as the "don't care" value indicating that the same transition takes place regardless of the value of that particular input.

The tables below were prepared by assuming that the controller obeys the inputs in the following order of priority: obstruction, the expected limit switch in (top/bottom), and direction buttons (up/down). If both buttons are pressed simultaneously the door reverses direction.

A row containing N 'X's is equivalent to 2^N rows with the 'X's replaced with the 2^N combinations of 1's and 0's. A quick check is to make sure that if there are N_{in} inputs the sum of the 2^Ns is $2^{N_{\text{in}}}$.

current state	input conditions					next state
	`obs`	`up`	`down`	`top`	`bottom`	
D	1	X	X	X	X	U
D	0	X	X	X	1	S
D	0	0	X	X	0	D
D	0	1	X	X	0	U
S	1	X	X	X	X	U
S	0	0	0	X	X	S
S	0	0	1	X	X	D
S	0	1	0	X	X	U
S	0	1	1	X	X	S
U	1	X	X	X	X	U
U	0	X	X	1	X	S
U	0	X	0	0	X	U
U	0	X	1	0	X	D

Note that in each state there is exactly one row (transition) that matches (would happen) any possible pattern (input). Anything else would be an ambiguous specification.

We'll use A and B as the state variables. We'll use the following state encoding (others are also possible):

state	A	B
U	1	0
D	0	1
S	0	0

The state transition table can be re-written using these encoding as:

current state		input conditions					next state	
A	B	`obs`	`up`	`down`	`top`	`bottom`	A'	B'
0	1	1	X	X	X	X	1	0
0	1	0	X	X	X	1	0	0
0	1	0	0	X	X	0	0	1
0	1	0	1	X	X	0	1	0
0	0	1	X	X	X	X	1	0
0	0	0	0	0	X	X	0	0
0	0	0	0	1	X	X	0	1
0	0	0	1	0	X	X	1	0
0	0	0	1	1	X	X	0	0
1	0	1	X	X	X	X	1	0
1	0	0	X	X	1	X	0	0
1	0	0	X	0	0	X	1	0
1	0	0	X	1	0	X	0	1

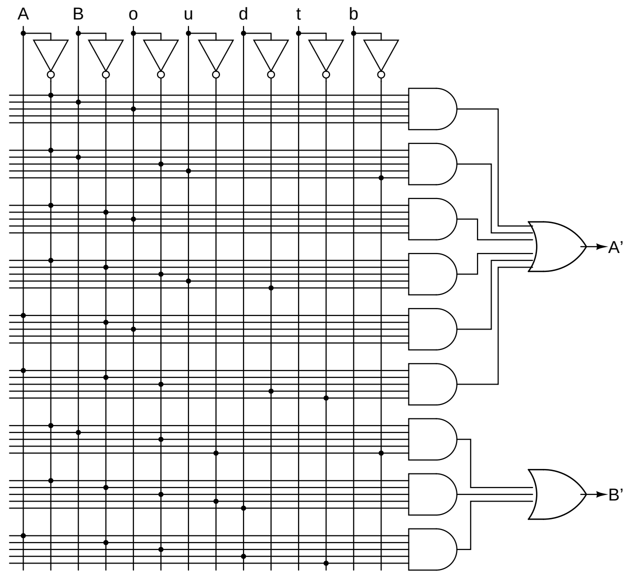
I'll use the boolean variables o, u, d, t, and b for `obs`, `up`, `down`, `top`, and `bottom` respectively.

The next-state equations are:

$$\begin{aligned} A' &= \overline{A}Bo + \overline{A}B\overline{o}u\overline{b} + \\ &\quad \overline{A}\overline{B}o + \overline{A}\overline{B}\overline{o}u\overline{d} + \\ &\quad A\overline{B}o + A\overline{B}\overline{o}\overline{d}\overline{t} \end{aligned}$$

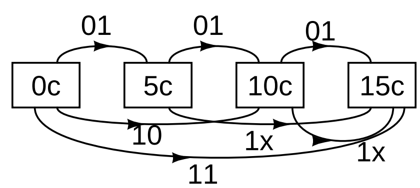
$$\begin{aligned} B' &= \overline{A}B\overline{o}\overline{u}\overline{b} + \\ &\quad \overline{A}\overline{B}\overline{o}\overline{u}d + \\ &\quad A\overline{B}\overline{o}d\overline{t} \end{aligned}$$

The schematic is:



Question 8

The controller inputs are the coin detector outputs (labelled X and Y). The controller output is the candy release signal (labelled R). The four states correspond to the possible sum of money deposited: 0, 5c, 10c, and 15c. The state transition diagram is:



The release is only turned on when the count of money reaches 15 cents. Tabular descriptions of the state transitions and outputs are:

current state	input conditions		next state
	X	Y	
0c	0	0	0c
0c	0	1	5c
0c	1	0	10c
0c	1	1	15c
5c	0	0	5c
5c	0	1	10c
5c	1	X	15c
10c	0	0	10c
10c	0	1	15c
10c	1	X	15c
15c	X	X	0c

state	R
0c	0
5c	0
10c	0
15c	1