1. Given the following code,

```
unsigned char m_and = x & mask;
unsigned char m_or = x | mask;
unsigned char m_land = x && mask;
unsigned char m_lor = x || mask;
```

Complete the following table, representing each value in hexadecimal:

	x	mask	m_and	m_or	m_land	m_lor
a.	0x01	0xff				
b.	0x08	0xf7				
c.	0xc7	0x00				
d.	0x0e	0x01				
e.	0x00	0x00				

2. Given the following code,

```
int x=-3;
long int y;
y=x;
```

What is the value of y in decimal? In hexadecimal?

Does padding on the left with the sign bit to increase precision change the value of a negative number? If so, why? If not, why not?

3. Complete the table below by inserting the relationship "<", "==", ">" in the "?" column to make the expression "x?y" true. Assume INT_MIN=-2,147,483,648, INT_MAX=2,147,483,647, and UINT_MAX=4,294,967,295

	x	?	y
a.	1		-2
b.	-1		(unsigned) -2
c.	2147483647		(int)2147483648
e.	2147483647		-2147483647-1
f.	(unsigned)2147483647		-2147483647-1
g.	INT_MAX+1		INT_MIN

4. Given the following:

```
int x=atoi(argc[2]);
int y=atoi(argc[3]);
unsigned int v=x; unsigned int w=y;
int s_sum; unsigned int u_sum;
s_sum=x+y;
u_sum=v+w;
if (s_sum==u_sum) printf("Unsigned and signed are the same\n");
else printf("Unsigned and signed are different\n");
```

What do you expect to get printed most often? Why?

Are there any values for argc[2] and argc[3] for which this will print "Unsigned and signed are different"?

What if you changed to "s_sum=x*y" and "u_sum=v*w"? Does the same thing happen as for addition?