Bitwise Operations Test

a = ?
 	
int orig = 0x7070;
int insert = 0x0005;
int a = orig ^ (insert << 4);
	a = ?
	
	int orig = 0xB0B0;
	int insert = 0x000A;
	int a = orig ^ (insert << 8);

OR = ?
 	
int orig = 0xB0B0;
int insert = 0x0003;
int a = orig | (insert << 8)
int b = orig | (insert << 12)
int OR = a | b;
	OR = ?
	
	int orig = 0xE0E0;
	int insert = 0x000B;
	int a = orig | (insert << 8)
	int b = orig | (insert << 12)
	int OR = a ^ b;

result = ?
 	
long value1 = 0x60067007;
long value2 = 0x60607070;
int result = (value1 << 4) ^ (value2 >> 12);
	result = ?
	
	long value1 = 0xC00C6006;
	long value2 = 0xC0C06060;
	int result = (value1 << 12) ^ (value2 >> 4);

result = ?
 	
long value1 = 885;
long value2 = 46;
int result = (value1 << 8) ^ (value2 >> 12);
	result = ?
	
	long value1 = 474;
	long value2 = 112;
	int result = (value1 << 4) ^ (value2 >> 8);

cupcake = ?
 	
int i = 0xA0A0;
int cupcake = i ^ (1 << 4);
	a = ?
	
	long testValue = 0xEE008800;
	int a = 0;
	if (testValue & (1 << 12))
	{
	a = 1;
	}
	else
	{
	a = 2;
	}

```
a = ?

long testValue = 0x40045005;
int a = 0;
if (testValue | testValue & (1 << 12))
{
a = 1;
}
else
{
a = 2;
}
```

```
a = ?, result = ?

long testValue = 0x50502020;
int a = 0;
if ((result = testValue ^ testValue | testValue
& (1 << 12)))
{
a = 1;
}
else
{
a = 2;
}
```