Technical HW 1 Solutions

Problem 1:

What are the two's complement representations for the following (decimal) numbers? Show your work.

The largest number here is +1024. For this in two's complement, we need 12 bits (11 bits for magnitude, 1 for the sign). Therefore, I will represent all of these with 12 bits.
The easiest way to do this is to find the nearest power of 2^n, and then work backwards. For negative numbers, get the positive value, then flip the bits and add 1.

a. $10 = 8 + 2 = 2^3 + 2^2 = 1010$
b. $436 = 2^8 + 2^7 + 2^5 + 2^4 + 2^2 = 0000110110100$
c. $1024 = 2^{10} = 010000000000$
d. $13 = 8 + 4 + 1 = 000000001101$, flip and add one,$-13 = 111111110011$
e. $1023 = 1024 - 1 = 001111111111$, flip and add one, $-1023 = 110000000001$
f. We did 1024 above so flip and add one,$-1024 = 110000000000$

Decimal Representation	**Binary Representation**
10	“000000001010"
436	“000110110100”
1024	“010000000000”
-13	“111111110011"
-1023	“110000000001"
-1024	“110000000000"

Problem 2:

Suppose I need to compute the series $f_n = f_{n-1}^2$. If the value $f_0 = 2$, what is the maximum n that can be stored in the following C++ data types, assuming that an int is 4 bytes, a long int is 8 bytes, and each byte stores 4 bits?

The series $2, 2^2, 2^{2^2}, 2^{2^{2^2}}, \cdots$ has an nth entry with value 2^{2^n}. Thus, the binary representation has a “1” in the 2^n place. If we have m bits, then we can store the largest n such that $2^n - 1 < m$ for signed values (so $n < \log_2(m+1)$), and $2^n < m$ (so $n < \log_2(m)$) for unsigned values. Be careful! This is a “less than” sign, not a “less than or equal” sign!

a. int (4 bytes * 4 bits/byte = 16 bits - 1 sign = 15 bits): log2(15) = 3.91
   thus **n=3**.

b. long (8 bytes * 4 bits/byte = 32 bits - 1 sign = 31 bits): log2(31)=4.95
   thus **n=4**.

c. unsigned long (8 bytes * 4 bits/byte = 32 bits): log2(32) = 5, but need the LOWER integer,
   thus this is **n=4** also.

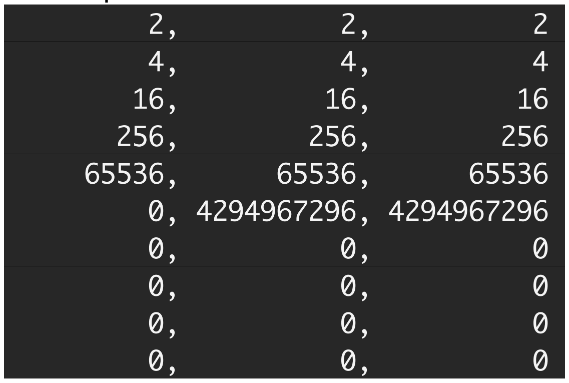
Problem 3:

Starting from one of the C++ programs in "ReviewCpp", write a C++ program to demonstrate Problem 2 on your computer. That is, implement the series using int, long int, and unsigned long int variables, and print out the value of the variable at each step of the series. Did you see what you expect? Why or why not?

Relevant piece of code:

```
  int i = 2;
  long l = 2;
  unsigned long ul = 2;
  for ( int p = 0; p < 10; ++p ) {
    std::cout << std::setw(10) << i << ", " << std::setw(10) << l <<
", " << std::setw(10) << ul << std::endl;
    i = i*i;
    l = l*l;
    ul = ul*ul;
  }
```

The output here is:



```
         2,          2,          2
         4,          4,          4
        16,         16,         16
       256,        256,        256
     65536,      65536,      65536
         0, 4294967296, 4294967296
         0,          0,          0
         0,          0,          0
         0,          0,          0
         0,          0,          0
```

We see that n = 4, n = 5, n = 5 for int, long, and unsigned int respectively. This is because the values we used for the size of int are not actually what are used, these are “short int” values. There are also different values you can get on different machines due to the standard only specifying the MINIMUM value.