4. In order to encode, or decode, a message using the RSA cryptosystem, you need three integers:

   (a) The *modulus* n, which is publicly known large integer, with typically hundreds of bits. Typically, $n = pq$, where p and q are large primes which are kept secret.

   (b) The *message* m, which is a number in the range $1 \ldots n-1$.

   (c) The *exponent* e, a positive integer, which could be fairly large as well, but never more than $n-1$.

   The message is encoded by computing $c = m^e (\text{mod } n)$.

   The message is then decoded by computing $m = c^d (\text{mod } n)$, where d is can be computed from e, p, and q, and is secret since p and q are secret.

   Here is some pseudocode which computes $c = m^e (\text{mod } n)$.

```
a = m;
b = e;
c = 1;
while (b > 0){
  if(b is odd)
    c = c*a % n;
  a = a*a % n;
  b = b/2; // Truncated division, as in C++
}
// The output is c.
```

Clue

To understand it, you need to try some numbers. Let $m = 2$, $e = 21$, $n = 19$. (You can directly compute $2^{21} = 2097152$, and $2097152 \% 19 = 8$. However, with realistic sized numbers, you couldn't.)

With those inputs, here are the values of the variables of the code before and after each iteration.

c	a	b
1	2	21
2	4	10
2	16	5
13	9	2
13	5	1
8	6	0