Intro/recap:
I set up the totient function as follows: I said that we needed to find some number phi(m) such that a^phi(m) = 1 mod m. I defined phi(m) as the number of numbers of units in the set Z_m, and reiterated that this was the number of numbers that are less than m and relatively prime to m.

What's left?
 - We need to prove that a^phi(m) = 1.
 - We need to compute phi(m)

Proving a^phi(m) = 1 if a is a unit:
We're going to think about what happens when we multiply things by higher and higher powers of a. Here's the set of units, and here's a:

```
{       a                                           }
```

What happens when we multiply multiply a and a? We get another unit (what's its inverse?).

```
           multiply by a
         -------------
        /             \
{       a             a^2                           }
```

Same thing with any other unit b:

```
           multiply by a
         -------------
        /             \
{       a      b      a^2          ab               }
               \                  /
                ------------------
                  multiply by a
```

First of all, what happens when we raise a to higher and higher powers? Well, there's only so many units, so there must be a loop. a^(x + y) = a^x. Well, since a is a unit, we can multiply by a^-x and get a^y = 1, so a^(y+1) = a. If we pick the smallest such power y, then we have this picture:

```
         -----------------------<<-------------------
        /                                            \
       | ----->-------   --->----   ------->--------  |
       |/             \ /       \ /                \ |
{       a        b    a^2        a^3    ab           a^(y-1)  }
                  \                      /
                   --------->>----------
```

And none of the a^ns are the same (otherwise y isn't the smallest!). We can also think about where the ba^ns go as we multiply by a:

```
         ---------------------------<<---------------------
        /                                                  \
       | ----->-----------   --->----   -----------------   |
       |/                 \ /       \ /                  \ |
{       a  a^y-1 ->- b    a^2        a^3    ab ->- a^2b  a^(y-1)  }
             |        \                      /       |
             |         --------->>----------         |
              \                                      /
               -----------------<<------------------
```

None of the elements in this picture can be the same. The b's can be the same as each other (small proof on the side), and the b's can't be the same as the a's (small proof on the side).

This might not be all the units of course, but if there's some c that we haven't drawn yet, it will be in its own cycle:

```
         ---------------------------<<---------------------
        /                                                  \
       | ----->-----------   --->----   -----------------   |              -------------------<<---------------
       |/                 \ /       \ /                  \ |             /                                    \
{       a  a^y-1 ->- b    a^2        a^3    ab ->- a^2b  a^(y-1)         c  -->>-- ac -->>-- a^2c -- ... -- a^(y-1)c  }
             |        \                      /       |
             |         --------->>----------         |
              \                                      /
               -----------------<<------------------
```

And c's cycle can't overlap with the other two.

So we've partitioned the entire set of units into these cycles. Each cycle contains y elements, and everything is in one of the cycles. So y must divide the total number of elements. The total number of elements is phi(m). So y divides phi(m). So raising a to the phi(m) means going around the loop phi(m)/y times, which gets us back to a.

QED.

Computing phi(m):

We already saw that phi(p) = p-1 if p is prime. We need to compute phi(pq) where p and q are distinct primes. We can do this by listing all the numbers and crossing off the non-units:

```
{  0    1    2    3 ... (p-1)       ---
   p  p+1  p+2  p+3 ... 2p-1         |
  2p 2p+1 2p+2 2p+3 ... 3p-1        q rows
            ...                      |
(q-1)p              ... qp-1}       ---

|--------- p columns -------|
```

Clearly the whole left hand column are not coprime with pq, and there are q of them. Everything else is coprime with p, so the only thing we have to worry about are the multiples of q. By the same picture, there are p multiples of q. The only overlap between the two is 0 (or pq if you prefer). So we have pq total elements, minus p multiples of q, minus q multiples of p, but plus one because we double counted zero.
pq - p - q + 1 = p(q-1) - (q-1) = (p-1)(q-1)

Summary of RSA:
The recipient publishes pq and k. The sender transmits a^k mod pq. The recipient computes phi(pq), and k^-1 mod phi(pq) (using the homework). He then computes (a^k)^(k^-1). kk^-1 = 1 + x*phi(pq). So a^{kk^-1} = a^(1 + x*phi(pq)) = a*a^(x*phi(pq)) = a.