Finishing proof of Euler's theorem (v2)

**Definition:** \([a]_m\) is a unit if there exists \([a]^{-1}\) with \([a][a]^{-1} = [1]\)

**Definition:** \(\mathbb{Z}_m^*\) is the set of units of \(\mathbb{Z}_m\).

**Claim:** if \([a]_m\) is a unit, then \([a]^{\phi(m)} = [1]\)

**Proof:** Choose arbitrary \(a\) and \(m\) with \([a]_m\) a unit. Draw the elements of \(\mathbb{Z}_m^*\) with an arrow from each \([b]\) to \([a][b]\). Various things can't happen:

- If \([b][a] = [c]\) and \([b][b] = [d]\), then \([c][c] = [d]\)
- \([a][b][c] = [b][a][c]\)
- So \([c][c] = [d][d]\)
- (mult. is well-defined)

Thus, if \([b][a] = [c]\) and \([a][c][c] = [d]\), then \([b][b] = [c]\)

**Question:** why not?

\[\begin{align*}
\text{if } [b][a] &= [c] \\
\text{and } [b][b] &= [d]\text{ then} \\
\text{then } [c][c] &= [d]
\end{align*}\]

... (further)

**Proof:** 

- \([b][a][c] = [b][a][c]^{-1} = [b][b]^{-1}[a]^{-1} = [1]^{-1}\)
- \((b)[a][c]^{-1}(c)[a] = (b)[b]^{-1}[a][c]^{-1}\)
- \(= [c][d][d] = [d][d][d] = [d][1][1] = [d][1] = [c][1]\)

... (further)

**Proof:** 

- \([b][b] = [c] = [d]\)
- \(a)[c][c] = [d]\)
- \(\text{assume } [a][b] = [c]\) and \([a][c] = [d]\)
- \(\text{then } [b][b] = [c]\)
- \(\text{well } [a][c][c] = [c][1][c]\)
- \(\text{so } c[1][1] = [c][1][1] = [c][1]\)
So words form cycles $[a^l \cdots a^2 a^1]$.

All loops have to be the same length.

In total (elements in picture) $\ell(m)$ total, so $\psi(m) = n \ell$.

To find $[a]^{\psi(m)}$, start at $[1]$, take $\psi(m)$ steps, and go around loop $n$ times.

So end at $[c^l \cdots c]$. 
Operations on modular numbers:

- \([a + b] := [a] + [b]\) is well defined
- \([a][b] := [ab]\) is well defined
- \([a]^{-1}\) (if \([a]\) is a unit) is well-defined
- \([a]^n := [a][a][a] \cdots [a] = [a^n]\) is well-defined
  \(n\) times
- \([a][b] := [a^b]\) is \textbf{not} well-defined
- \([a][b]_{\varphi(m)} := [a^b]_m\) is well-defined
Public key cryptography (RSA)

\[ a \rightarrow n \]
\[ b \rightarrow \lambda \]
\[ c \rightarrow x \]
\[ d \rightarrow a \]

Sender
\[ \text{secret} \]
\[ \text{msg} \]
\[ \text{secret} \]
\[ \text{msg} \rightarrow \text{Enc} \]
\[ \text{looks like random garbage} \]
\[ \text{Dec} \rightarrow \text{msg} \]

recipient
\[ \text{secret} \]
\[ \text{secret} \]
\[ \text{Dec} \rightarrow \text{msg} \]

Sender
\[ \text{msg} \]
\[ \text{public key (pk)} \]
\[ \text{Enc} \rightarrow \text{cypher text} \]
\[ \text{recipient} \]
\[ \text{public key, private key} \]
\[ \text{Dec} \rightarrow \text{msg} \]
Key assumptions:

- hard to factor large numbers.
- unit mod m.
- large primes.

\[ C(m) = (p-1)(q-1) \]

If you know \( k \) and \( C(m) \), can find \( [k]^{-1} \) using Bezout coeff.

\[ \sqrt{C} = k \sqrt{msg} \equiv msg. \]

\[ [C]_m = [msg]_m \]

\[ [C]_{k^{J(m)}} = [msg]_{k^{J(m)}} \]
Fast exponentiation

Computing $a^k$ naively takes $k - 1$ multiplications. Can we do better?

We'll cover on Monday. Here's the trick:

Instead of finding $a \cdot a$ and $(a \cdot a) \cdot a$ and $(a \cdot a \cdot a) \cdot a$

$\vdots$

Instead, find $a^4 = (a \cdot a) \cdot (a \cdot a)$ and $\cdots$

$4$ mults.

Then write $k$ as a sum of powers of $2$ (binary).

E.g. if $k = 23 = 16 + 4 + 2 + 1$

Then $a^k = a^1 \cdot a^4 \cdot a^2 \cdot a$