

Quiz 2 will be on Monday, April 9th, in class. It will cover homeworks 6, 7, and 8.

RSA example:

Generate PK, SK:

1. $p = 5, q = 11$
2. $n = 5(11) = 55, \phi(n) = (5 - 1)(11 - 1) = 40$
3. Find e such that $\gcd(e, \phi(n)) = 1$ (we want $1 \leq e < \phi(n)$)

We happen to know that

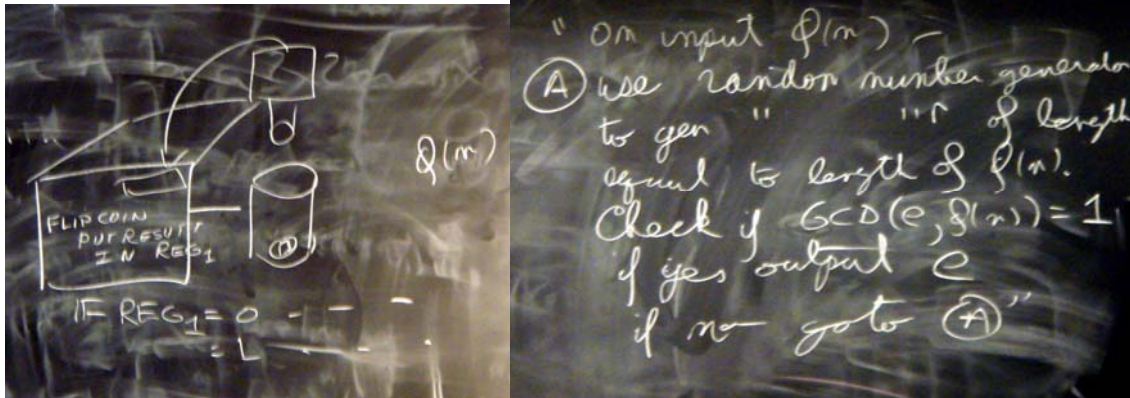
$$\frac{1}{\ln^2 n} \leq \frac{|\{e : 1 \leq e \leq \phi(n) \& \gcd(e, \phi(n)) = 1\}|}{|\{e : 1 \leq e \leq \phi(n)\}|}$$

Algorithm FINDe:

On input $\phi(n)$:

A: use random number generator to generate a random number e of length equal to the length of $\phi(n)$. Check if $\gcd(e, \phi(n)) = 1$ using Euclid's algorithm. If yes, output e , if no, goto A.

Random number generator:



Euclid's algorithm:

Example on 1547 and 560:

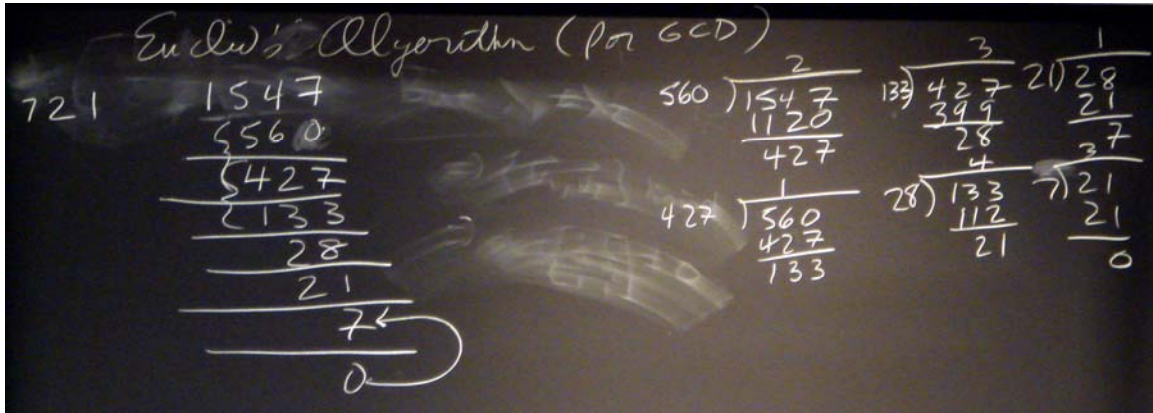
Write down the larger number above the smaller. Divide larger number by the smaller one

$$\frac{1547}{560} = 2 + \frac{427}{560}. \text{ Write down the remainder below the two previous numbers, and repeat}$$

on the two lowest numbers.

$$\frac{560}{427} = 1 + \frac{133}{427}, \text{ and so on... } \frac{427}{133} = 3 + \frac{28}{133} \dots \frac{133}{28} = 4 + \frac{21}{28} \dots \frac{28}{21} = 1 + \frac{7}{21} \dots \frac{21}{7} = 3.$$

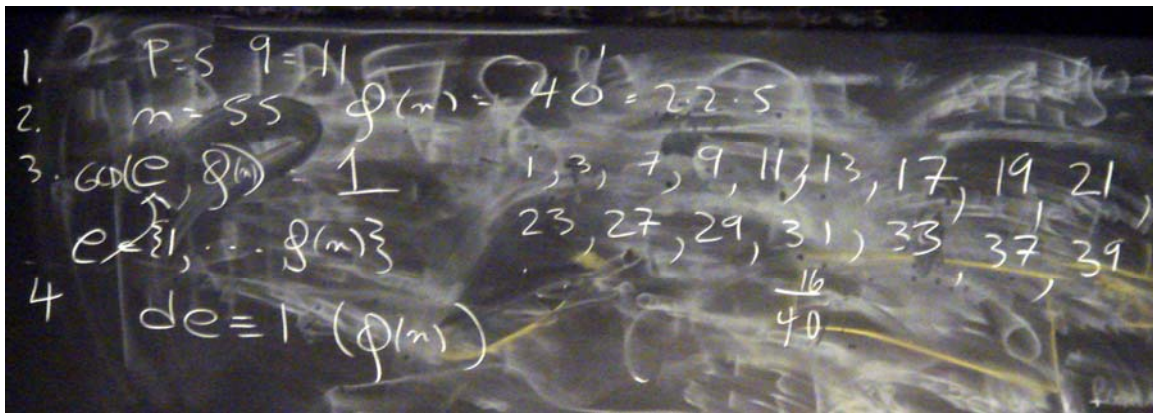
The number you wrote down last, before the remainder was 0, is the gcd of the two inputs. In this case, the gcd is 7.



Lamé (1800s) showed that Euclid's algorithm runs in polynomial time (though in a different mathematical language). This is the first known analysis of an algorithm's running time.

It follows that FINDe takes expected polynomial time.

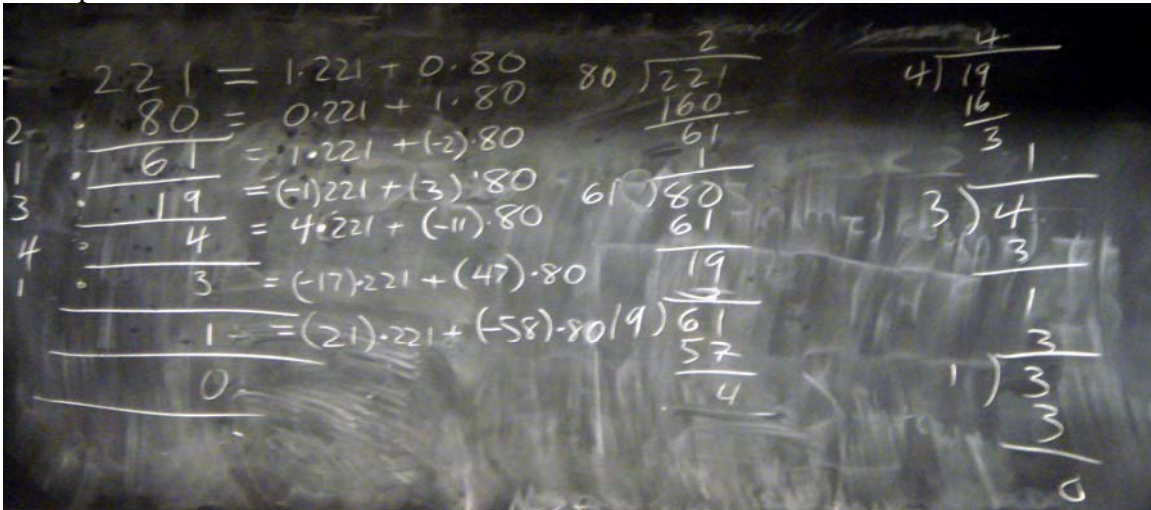
Let's say FINDe picked $e = 29$.



4. Find d such that $de \equiv 1 \pmod{\phi(n)}$

Extended Euclid's algorithm keeps track of the quotient to find a linear combination of the inputs that is equal to the gcd:

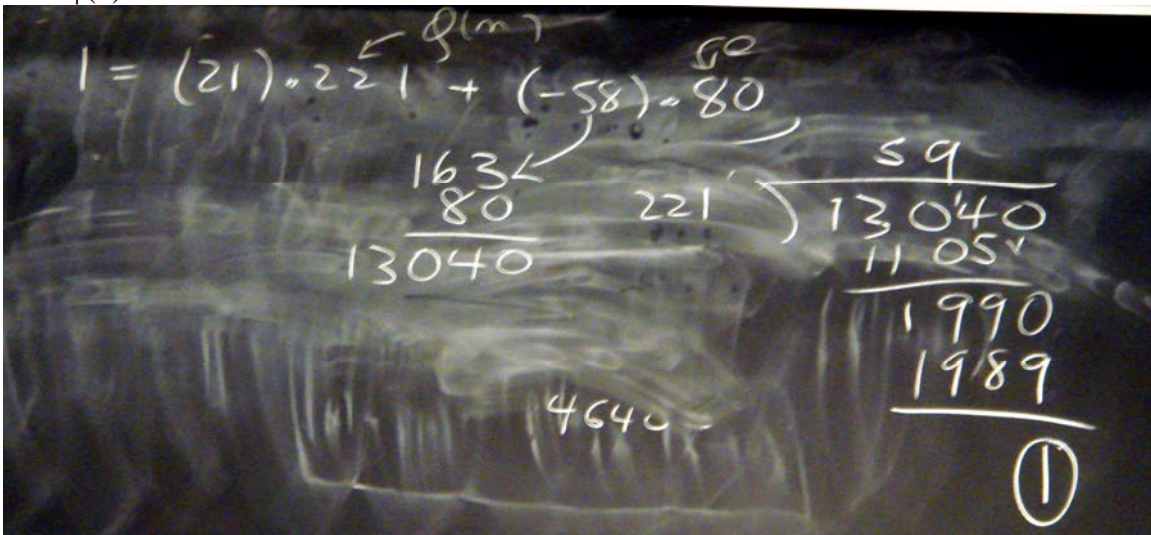
Example on 221 and 80:



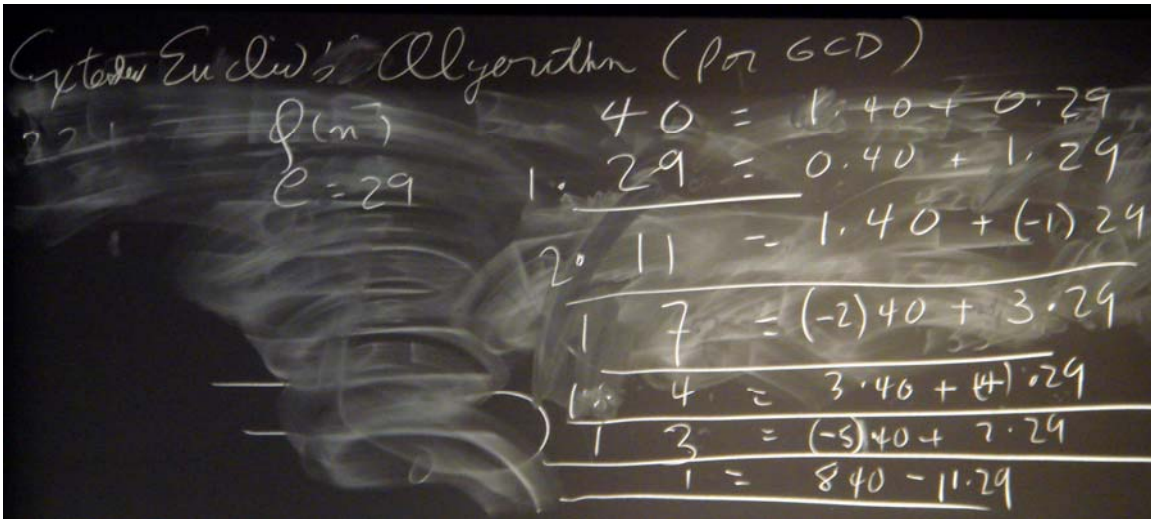
So we get

$1 = (21) 221 + (-58) 80$. Suppose our $\phi(n) = 221$ and our choice of $e = 80$. To find d , we execute the extended Euclid's algorithm to find that $1 = (21) 221 + (-58) 80$. Because the -58 is negative, we add $\phi(n)$ to it (if it were positive, we'd use it as is). $\phi(n) - 58 = 163$.

Now observe that $163 (80) = 13040$, and $\frac{13040}{221} = 59 + \frac{1}{221}$, so if $d = 163$, then $de \equiv 1 \pmod{\phi(n)}$.



Now back to our example. We had $\phi(n) = 40$ and $e = 29$. We run extended Euclid's algorithm (we actually could just do this in step 3 to get e and d simultaneously):



So we get $1 = (8) 40 + (-11) 29$. Again, we add 40 to -11 to get 29 (just a coincidence here that we got 29 again). Note that $29(29) = 841$, and $\frac{841}{29} = 21 + \frac{1}{29}$, so if $d = 29$, then $de \equiv 1 \pmod{\phi(n)}$. So if $e = 29$ then $d = 29$.

5. Set $PK = \langle 29, 55 \rangle$

6. Set $SK = \langle 29, 55 \rangle$

It is just a coincidence that they are the same here, and in general, they will not be the same!

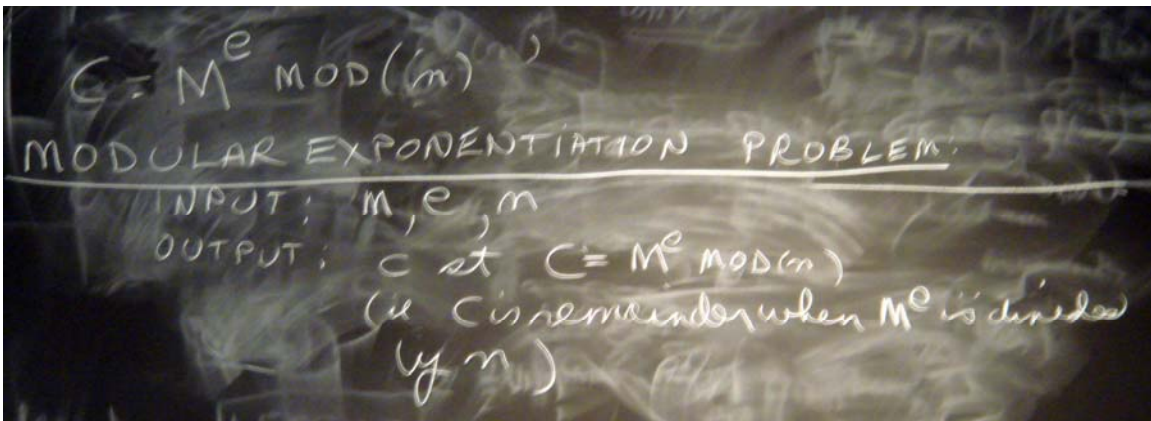
Encrypt algorithm:

We wish to find $m^e \pmod n$.

Modular Exponentiation Problem:

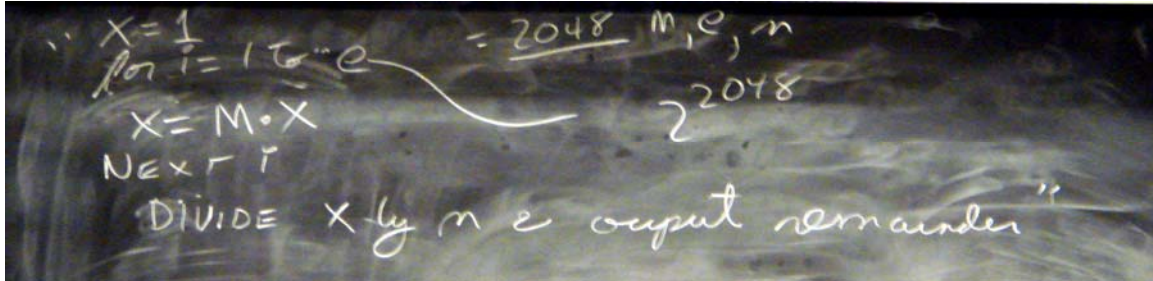
Input: m, e, n

Output: c such that $c = m^e \pmod n$ (i.e. c is the remainder of $\frac{m^e}{n}$).



Algorithm 1:

```
x = 1
for i = 1 to e
    x *= m
divide x by n and output the remainder
```



Algorithm 1 is not polynomial!

Algorithm 2:

We can use repeated squaring. For $e \approx 2^k$, find m^{2^i} for $i = 1, 2, \dots, k$ and multiply the m^{2^i} for all i such that 2^i is in the binary representation of e .

Algorithm 2 is still not polynomial! Note that m^e , for $m, e \approx 2^k$ is HUGE. $m^e \approx (2^k)^{2^k}$ (we can't even write that many digits down!)