Last class:

- Modular arithmetic
- Modular exponentiation:
  for n-bit x, y, N, compute $x^y \bmod N$
  in $O(n^3)$ time using "repeated squaring" idea.
- Inverses: $x \equiv a^{-1} \bmod N$ means $ax \equiv 1 \bmod N$
  $a^{-1} \bmod N$ exists iff $\underbrace{\gcd(a,N)=1}_{a \,\&\, N \text{ are relatively prime.}}$
- Can check $\gcd(a,N)$ using Euclid's alg. in $O(n^3)$ time
  & can compute $a^{-1} \bmod N$ using Extended Euclid alg.
  (if it exists)

This class:

- Fermat's little theorem.
  - used for RSA protocol
    & for primality testing.
- RSA cryptosystem.

Fermat's Little Theorem:

If p is prime then for all $a \not\equiv 0 \bmod p$,

$$a^{p-1} \equiv 1 \bmod p$$

Proof:

Let $S = \{1, 2, 3, \ldots, p-1\}$

Let $S' = aS \bmod p$

$$= \{1 \times a \bmod p, 2 \times a \bmod p, 3 \times a \bmod p, \ldots, (p-1) \times a \bmod p\}$$

(Example: $p = 7$, $a = 3$, then

$S = \{1, 2, \ldots, 6\}$, $S' = \{3, 6, 2, 5, 1, 4\}$)

note in this example $S = S'$, just different order.
Let's prove that's true in general.

Claim: $S = S'$

Proof: We'll prove:

1. Elements of S' are distinct mod p.
2. None of S' is $0 \bmod p$.

Thus, S' has $p-1$ non-zero elements, & therefore it must be the same as S, so that'll prove the claim we just need to prove 1 & 2.

1. Elements of S' are distinct:

Suppose for $i \neq j$ where $1 \leq i, j \leq p-1$,

$ai \equiv aj \mod p$ (So i^{th} & j^{th} elements are the same)

Since p is prime then we know

$a^{-1} \mod p$ exists

thus,

$aia^{-1} \equiv aja^{-1} \mod p$

$i \equiv j \mod p$ since $aa^{-1} \equiv 1 \mod p$

$\times$

2. Suppose $ai \equiv 0 \mod p$

then $aia^{-1} \equiv 0a^{-1} \mod p$

$i \equiv 0 \mod p$

So only the 0^{th} element is 0, but there is no 0^{th} element. $\blacksquare$

Since $S \equiv S'$ mod p (just different order)

thus: $$\prod_{z \in S} z \equiv \prod_{z' \in S'} z' \mod p$$

$$(1)(2)(3)\cdots(p-1) \equiv (a)(1)(a)(2)\cdots(a)(p-1) \mod p$$

Since p is prime, $1^{-1}, 2^{-1}, 3^{-1}, \dots, (p-1)^{-1} \mod p$ exists.

$$\underbrace{(1)(1^{-1})}_{1}\underbrace{(2)(2^{-1})}_{1}\cdots\underbrace{(p-1)((p-1)^{-1})}_{1} \equiv a^{p-1}\underbrace{(1)(1^{-1})}_{1}\underbrace{(2)(2^{-1})}_{1}\cdots\underbrace{(p-1)(p-1)^{-1}}_{1} \mod p$$

$$1 \equiv a^{p-1} \mod p$$

∎

We'll use Fermat's Little Theorem to test if a number N is prime.

For all $a \in \{1, \dots, N-1\}$,

$$a^{N-1} \equiv 1 \mod N \text{ if } N \text{ is prime.}$$

& if N is composite & not "Pseudoprime"

then for at least half of $a \in \{1, \dots, N-1\}$,

$$a^{N-1} \not\equiv 1 \mod N.$$

Suppose N is not prime, how does Fermat's Little Theorem generalize. We'll use it for $N = pq$ for primes p & q.

For integer $N \geq 1$,

let $\phi(N) = \#$ integers b in $\{1, \dots, N\}$ where $\gcd(b, N) = 1$

$= \#$ of positive integers up to N that are relatively prime to N.

$\phi(N)$ is called Euler's totient function.

Euler's Theorem: For any N, a where $\gcd(a, N) = 1$,

$$a^{\phi(N)} \equiv 1 \bmod N.$$

If $N = p$ for prime p then $\phi(N) = p-1$ so we get Fermat's little theorem.

For primes p, q, $\phi(pq) = (p-1)(q-1)$.
Why? $1, \dots, pq$ has pq numbers, then crossout the p multiples of q & the q multiples of p & we crossed out pq twice: $pq - p - q + 1 = (p-1)(q-1)$.

– For prime p, take bc where $bc \equiv 1 \mod p-1$

this means $bc = 1 + k(p-1)$ for some integer k.

Note, by Fermat's Little Theorem, $a^{p-1} \equiv 1 \mod p$

& thus, $a^{bc} \equiv (a)\cancelto{1}{(a)^{k(p-1)}} \equiv a \mod p$

– For primes p & q, take de where $de \equiv 1 \mod (p-1)(q-1)$,

and thus $de = 1 + k(p-1)(q-1)$ for integer k.

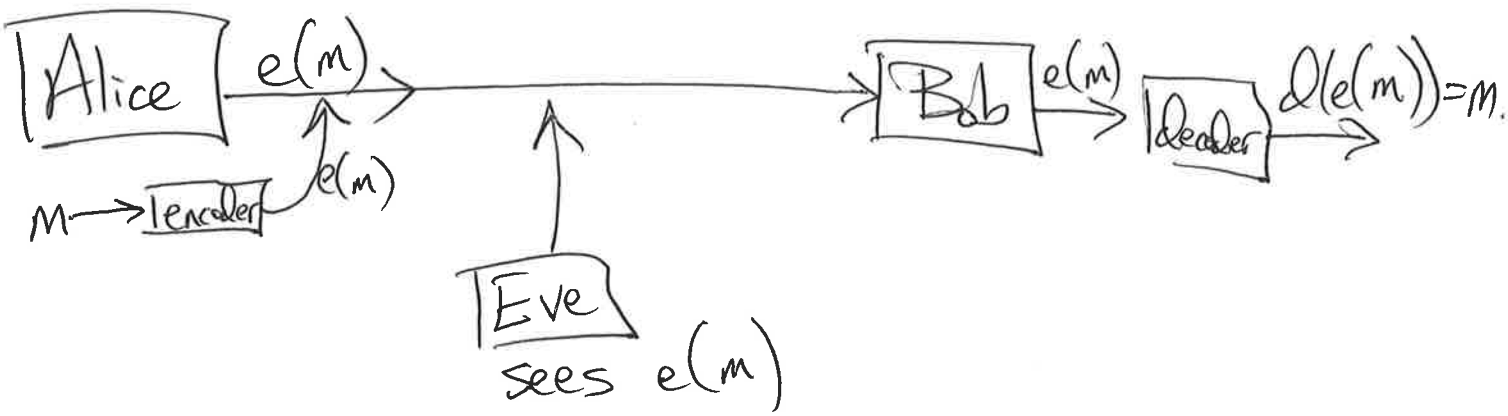
By Euler's Thm., for a where $\gcd(a, pq) = 1$,

$$a^{(p-1)(q-1)} \equiv 1 \mod pq$$

& thus, $a^{de} \equiv (a)\left(a^{(p-1)(q-1)}\right)^k \equiv a \mod pq$

Cryptography setting:

Alice has a message m (view as a n-bit #)
that she wants to send to Bob
but Eve sees the message sent.



Public-key cryptography:

Bob publishes a public key (N, e)
Anyone sending a message to Bob uses
Bob's public key to encrypt
& only Bob can decrypt.

RSA protocol: Let n = HUGE # of bits $\approx 4{,}000$

Bob:

1) Chooses 2 n-bit random primes p & q

HOW?

- Choose random n-bit number & check if its prime.

  Prob. it's prime $\approx \frac{1}{n}$.

  If it is prime use it & if not repeat

2) Bob finds e relatively prime to $(p-1)(q-1)$

   Try $e = 3, 5, 7, 11, 13, \ldots$

   & check if $\gcd(e, (p-1)(q-1)) = 1$.

3) Let $N = pq$.

4) Bob publishes his public key (N, e).

5) Bob computes his private key:

$$d \equiv e^{-1} \bmod (p-1)(q-1)$$

using Extended-Euclid algorithm.

Alice: To send message m to Bob:

1. Looks up Bob's public key (N, e)
2. Computes $y \equiv m^e \bmod N$

   using fast modular exponenation alg. (i.e., repeated squaring)
3. Sends y to Bob.

Bob:

1. Receives y & decrypts using

$$m \equiv y^d \bmod N$$

Why is this $\nearrow$?

B/c $de \equiv 1 \bmod (p-1)(q-1)$ so $de = 1 + k(p-1)(q-1)$.

& thus, $y^d \equiv (m^e)^d \equiv (m)\left(m^{(p-1)(q-1)}\right)^k \equiv m \bmod N$

for m where $\gcd(m, N) = 1$

& if $\gcd(m, N) > 1$ then it's still true (Using Chinese Remainder Thm.)

but this m can factor N (so shouldn't use)

Key assumption:

Given N, e & y (where $y \equiv m^e \bmod N$)
it is computationally difficult to determine m.

Natural way is to factor N to get p & q
then we can compute $(p-1)(q-1)$ &
find $d \equiv e^{-1} \bmod (p-1)(q-1)$.
But how to factor N?

Other issues:

- if $e=3$ (or e is small) then need to make sure $m^e > N$ otherwise mod N isn't doing anything.
  Solution: Pad m by choosing random r & sending $r+e$ & r.

- if send same m to e or more people (who use same e but diff. N) then if we see all e encrypted messages we can decrypt. So need to send a unique message each time.