To homework or not to homework.

Form extended.

There is NOT a “scoring bump” nor is there a “scoring detriment” for doing homework option oversus non-homework option.

We grade by making buckets according to quality of exams on exam totals.

Determines number of grades at each level. Then re-sort homework students only. So only their grades are affected.

Difference is about how you want to use your time and effort to do your best learning.

Today

Finish Euclid.

Bijection/CRT/Isomorphism.

Fermat's Little Theorem.

More divisibility

Notation: $d|x$ means “d divides x” or
$x = kd$ for some integer k.

Lemma 1: If $d|x$ and $d|y$ then $d|y$ and $d| \mod(x,y)$.

Proof:

$$\begin{aligned}\mod(x,y) &= x - \lfloor x/y \rfloor \cdot y \\ &= x - \lfloor s \rfloor \cdot y \quad \text{for integer } s \\ &= kd - s\ell d \quad \text{for integers } k,\ell \text{ where } x = kd \text{ and } y = \ell d \\ &= (k - s\ell)d\end{aligned}$$

Therefore $d| \mod(x,y)$. And $d|y$ since it is in condition. □

Lemma 2: If $d|y$ and $d| \mod(x,y)$ then $d|y$ and $d|x$.
Proof...: Similar. Try this at home. □ish.

GCD Mod Corollary: $\gcd(x,y) = \gcd(y, \mod(x,y))$.
Proof: x and y have **same** set of common divisors as x and $\mod(x,y)$ by Lemma 1 and 2.
Same common divisors $\implies$ largest is the same. □

Euclid's algorithm.

GCD Mod Corollary: $\gcd(x,y) = \gcd(y, \mod(x,y))$.

Hey, what's $\gcd(7,0)$? 7 since 7 divides 7 and 7 divides 0
What's $\gcd(x,0)$? x

```
(define (euclid x y)
  (if (= y 0)
    x
    (euclid  y (mod x y))))  ***
```

Theorem: (euclid x y) $= \gcd(x,y)$ if $x \geq y$.

Proof: Use Strong Induction.
Base Case: $y = 0$, “x divides y and x”
$\implies$ “x is common divisor and clearly largest.”
Induction Step: $\mod(x,y) < y \leq x$ when $x \geq y$

call in line (***) meets conditions plus arguments “smaller”
and by strong induction hypothesis
computes $\gcd(y, \mod(x,y))$
which is $\gcd(x,y)$ by GCD Mod Corollary. □

Excursion: Value and Size.

Before discussing running time of gcd procedure...

What is the value of 1,000,000?

one million or 1,000,000!

What is the “size” of 1,000,000?

Number of digits in base 10: 7.

Number of bits (a digit in base 2): 21.

For a number x, what is its size in bits?

$$n = b(x) \approx \log_2 x$$

Euclid procedure is fast.

Theorem: (euclid x y) uses $2n$ ”divisions” where $n = b(x) \approx \log_2 x$.

Is this good? Better than trying all numbers in $\{2, \ldots y/2\}$?

Check 2, check 3, check 4, check 5 . . . , check $y/2$.

If $y \approx x$ roughly y uses n bits ...
2^{n-1} divisions! Exponential dependence on size!

101 bit number. $2^{100} \approx 10^{30} =$ “million, trillion, trillion” divisions!

$2n$ is much faster! .. roughly 200 divisions.

Poll.

Assume $\log_2 1,000,000$ is 20 to the nearest integer.
Mark what's true.

(A) The size of 1,000,000 is 20 bits.
(B) The size of 1,000,000 is one million.
(C) The value of 1,000,000 is one million.
(D) The value of 1,000,000 is 20.

(A) and (C).

Poll

Which are correct?

(A) gcd(700,568) = gcd (568,132)
(B) gcd(8,3) = gcd(3,2)
(C) gcd(8,3) = 1
(D) gcd(4,0) = 4

Algorithms at work.

Trying everything

Check 2, check 3, check 4, check 5 ..., check $y/2$.

"(gcd x y)" at work.

```
euclid(700,568)
  euclid(568, 132)
    euclid(132, 40)
      euclid(40, 12)
        euclid(12, 4)
          euclid(4, 0)
            4
```

Notice: The first argument decreases rapidly.
At least a factor of 2 in two recursive calls.

(The second is less than the first.)

Runtime Proof.

```
(define (euclid x y)
 (if (= y 0)
     x
    (euclid  y (mod x y))))
```

Theorem: (euclid x y) uses $O(n)$ "divisions" where $n = b(x)$.

Proof:

Fact:
First arg decreases by at least factor of two in two recursive calls.

After $2\log_2 x = O(n)$ recursive calls, argument x is 1 bit number.
One more recursive call to finish.
1 division per recursive call.
$O(n)$ divisions.

□

Runtime Proof (continued.)

```
(define (euclid x y)
 (if (= y 0)
     x
    (euclid  y (mod x y))))
```

Fact:
First arg decreases by at least factor of two in two recursive calls.

Proof of Fact: Recall that first argument decreases every call.

Case 1: $y < x/2$, first argument is y
$\implies$ true in one recursive call;

Case 2: Will show "$y \geq x/2$" $\implies$ "$mod(x,y) \leq x/2$."

mod (x,y) is second argument in next recursive call,
and becomes the first argument in the next one.

When $y \geq x/2$, then

$\lfloor \frac{x}{y} \rfloor = 1$,

$\text{mod } (x,y) = x - y\lfloor \frac{x}{y} \rfloor = x - y \leq x - x/2 = x/2$

□

Poll

Mark correct answers.
Note: Mod(x,y) is the remainder of x divided by y.

(A) $\text{mod } (x,y) < y$
(B) If euclid(x,y) calls euclid(u,v) calls euclid (a,b) then $a <= x/2$.
(C) euclid(x,y) calls euclid (u,v) means u = y.
(D) if $y > x/2, mod(x,y) < y/2$
(E) if $y > x/2, mod(x,y) = (y - x)$

(D) is not always true.

Finding an inverse?

We showed how to efficiently tell if there is an inverse.

Extend euclid to find inverse.

Euclid's GCD algorithm.

```
(define (euclid x y)
 (if (= y 0)
     x
    (euclid  y (mod x y))))
```

Computes the gcd(x,y) in $O(n)$ divisions. (Remember $n = \log_2 x$.)

For x and m, if gcd$(x,m) = 1$ then x has an inverse modulo m.

Multiplicative Inverse.

GCD algorithm used to tell **if** there is a multiplicative inverse.

How do we **find** a multiplicative inverse?

Extended GCD

Euclid's Extended GCD Theorem: For any x,y there are integers a,b such that
$ax + by = d$ where $d = gcd(x,y)$.

"Make d out of sum of multiples of x and y."

What is multiplicative inverse of x modulo m?

By extended GCD theorem, when gcd$(x,m) = 1$.

$$ax + bm = 1$$
$$ax \equiv 1 - bm \equiv 1 \pmod{m}.$$

So a multiplicative inverse of $x \pmod{m}$!!
Example: For $x = 12$ and $y = 35$, gcd$(12,35) = 1$.

$(3)12 + (-1)35 = 1$.

$a = 3$ and $b = -1$.
The multiplicative inverse of $12 \pmod{35}$ is 3.

Check: $3(12) = 36 = 1 \pmod{35}$.

Make d out of multiples of x and y..?

```
gcd(35,12)
  gcd(12, 11)  ;;  gcd(12, 35%12)
    gcd(11, 1)  ;;  gcd(11, 12%11)
      gcd(1,0)
        1
```

How did gcd get 11 from 35 and 12?
$35 - \lfloor \frac{35}{12} \rfloor 12 = 35 - (2)12 = 11$

How does gcd get 1 from 12 and 11?
$12 - \lfloor \frac{12}{11} \rfloor 11 = 12 - (1)11 = 1$

Algorithm finally returns 1.

But we want 1 from sum of multiples of 35 and 12?

Get 1 from 12 and 11.
$1 = 12 - (1)11 = 12 - (1)(35 - (2)12) = (3)12 + (-1)35$
Get 11 from 35 and 12 and plugin.... Simplify. $a = 3$ and $b = -1$.

Extended GCD Algorithm.

```
ext-gcd(x,y)
  if y = 0 then return(x, 1, 0)
     else
         (d, a, b) := ext-gcd(y, mod(x,y))
         return (d, b, a - floor(x/y)  * b)
```

Claim: Returns (d,a,b): $d = gcd(a,b)$ and $d = ax + by$.
Example: $a - \lfloor x/y \rfloor \cdot b = 1 - \lfloor 35/12 \rfloor \cdot (-1) = 3$

```
ext-gcd(35,12)
  ext-gcd(12, 11)
    ext-gcd(11, 1)
      ext-gcd(1,0)
      return (1,1,0) ;; 1 = (1)1 + (0) 0
    return (1,0,1)   ;; 1 = (0)11 + (1)1
  return (1,1,-1)    ;; 1 = (1)12 + (-1)11
return (1,-1, 3)     ;; 1 = (-1)35 +(3)12
```

Extended GCD Algorithm.

```
ext-gcd(x,y)
  if y = 0 then return(x, 1, 0)
     else
         (d, a, b) := ext-gcd(y, mod(x,y))
         return (d, b, a - floor(x/y)  * b)
```

Theorem: Returns (d,a,b), where $d = gcd(a,b)$ and

$$d = ax + by.$$

Correctness.

Proof: Strong Induction.[1]
Base: ext-gcd$(x,0)$ returns $(d = x, 1, 0)$ with $x = (1)x + (0)y$.

Induction Step: Returns (d, A, B) with $d = Ax + By$
Ind hyp: **ext-gcd**$(y, \mod(x,y))$ returns (d,a,b) with
$d = ay + b(\mod(x,y))$

ext-gcd(x,y) calls **ext-gcd**$(y, \mod(x,y))$ so

$$\begin{aligned} d &= ay + b\cdot(\mod(x,y)) \\ &= ay + b\cdot(x - \lfloor \frac{x}{y} \rfloor y) \\ &= bx + (a - \lfloor \frac{x}{y} \rfloor \cdot b)y \end{aligned}$$

And ext-gcd returns $(d, b, (a - \lfloor \frac{x}{y} \rfloor \cdot b))$ so theorem holds! $\square$

[1] Assume d is $gcd(x,y)$ by previous proof.

Review Proof: step.

```
ext-gcd(x,y)
  if y = 0 then return(x, 1, 0)
     else
         (d, a, b) := ext-gcd(y, mod(x,y))
         return (d, b, a - floor(x/y)  * b)
```

Recursively: $d = ay + b(x - \lfloor \frac{x}{y} \rfloor \cdot y) \implies d = bx - (a - \lfloor \frac{x}{y} \rfloor b)y$

Returns $(d, b, (a - \lfloor \frac{x}{y} \rfloor \cdot b))$.

Hand Calculation Method for Inverses.

Example: gcd$(7,60) = 1$.
egcd(7,60).

$$\begin{aligned} 7(0) + 60(1) &= 60 \\ 7(1) + 60(0) &= 7 \\ 7(-8) + 60(1) &= 4 \\ 7(9) + 60(-1) &= 3 \\ 7(-17) + 60(2) &= 1 \end{aligned}$$

Confirm: $-119 + 120 = 1$

Note: an "iterative" version of the e-gcd algorithm.

Wrap-up

Conclusion: Can find multiplicative inverses in $O(n)$ time!

Very different from elementary school: try 1, try 2, try 3...
$2^{n/2}$

Inverse of $500,000,357$ modulo $1,000,000,000,000$?
≤ 80 divisions.
versus $1,000,000$

Internet Security.
Public Key Cryptography: 512 digits.
512 divisions vs.
$(10000000000000000000000000000000000000000000)^5$ divisions.

Internet Security: Soon.

Bijections

Bijection is **one to one** and **onto.**
Bijection:
$f : A \to B$.
Domain: A, Co-Domain: B.
Versus Range.
E.g. **sin** (x).
$A = B =$ reals.
Range is $[-1,1]$. Onto: $[-1,1]$.
Not one-to-one. **sin** $(\pi) =$ **sin** $(0) = 0$.

Range Definition always is onto.
Consider $f(x) = ax \mod m$.
$f : \{0, \ldots, m-1\} \to \{0, \ldots, m-1\}$.
Domain/Co-Domain: $\{0, \ldots, m-1\}$.

When is it a bijection?
When $gcd(a,m)$ is ....? ... 1.

Not Example: $a = 2$, $m = 4$, $f(0) = f(2) = 0 \pmod 4$.

Lots of Mods

$x = 5 \pmod 7$ and $x = 3 \pmod 5$.

What is $x \pmod{35}$?

Let's try 5. Not $3 \pmod 5$!
Let's try 3. Not $5 \pmod 7$!

If $x = 5 \pmod 7$
then x is in $\{5, 12, 19, 26, 33\}$.

Oh, only 33 is $3 \pmod 5$.
Hmmm... only one solution.

A bit slow for large values.

Simple Chinese Remainder Theorem.

My love is won. Zero and One. Nothing and nothing done.

Find $x = a \pmod m$ and $x = b \pmod n$ where gcd(m,n)=1.

CRT Thm: There is a unique solution $x \pmod{mn}$.
Proof (solution exists):
Consider $u = n(n^{-1} \pmod m)$.
$u = 0 \pmod n$ $\quad u = 1 \pmod m$
Consider $v = m(m^{-1} \pmod n)$.
$v = 1 \pmod n$ $\quad v = 0 \pmod m$
Let $x = au + bv$.
$x = a \pmod m$ since $bv = 0 \pmod m$ and $au = a \pmod m$
$x = b \pmod n$ since $au = 0 \pmod n$ and $bv = b \pmod n$
This shows there is a solution. $\square$

Simple Chinese Remainder Theorem.

CRT Thm: There is a unique solution $x \pmod{mn}$.

Proof (uniqueness):
If not, two solutions, x and y.

$(x - y) \equiv 0 \pmod m$ and $(x - y) \equiv 0 \pmod n$.
$\implies (x - y)$ is multiple of m and n
gcd$(m,n) = 1 \implies$ no common primes in factorization m and n
$\implies mn | (x - y)$
$\implies x - y \geq mn \implies x, y \notin \{0, \ldots, mn - 1\}$.
Thus, only one solution modulo mn. $\square$

Poll.

My love is won,
Zero and one.
Nothing and nothing done.

What is the rhyme saying?

(A) Multiplying by 1, gives back number. (Does nothing.)
(B) Adding 0 gives back number. (Does nothing.)
(C) Rao has gone mad.
(D) Multiplying by 0, gives 0.
(E) Adding one does, not too much.

All are (maybe) correct.
(E) doesn't have to do with the rhyme.
(C) Recall Polonius:
"Though this be madness, yet there is method in 't."

CRT:isomorphism.

For m, n, gcd$(m,n) = 1$.

$x \mod mn \leftrightarrow x = a \mod m$ and $x = b \mod n$
$y \mod mn \leftrightarrow y = c \mod m$ and $y = d \mod n$

Also, true that $x + y \mod mn \leftrightarrow a + c \mod m$ and $b + d \mod n$.

Mapping is "isomorphic":
corresponding addition (and multiplication) operations consistent with mapping.

Fermat's Theorem: Reducing Exponents.

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod p$,

$a^{p-1} \equiv 1 \pmod p$.

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p - 1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p - 1\}$ modulo p.

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

Since multiplication is commutative.

$$a^{(p-1)}(1 \cdots (p - 1)) \equiv (1 \cdots (p - 1)) \mod p.$$

Each of $2, \ldots (p - 1)$ has an inverse modulo p, solve to get...

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

$\square$

Poll

Which was used in Fermat's theorem proof?

(A) The mapping $f(x) = ax \mod p$ is a bijection.
(B) Multiplying a number by 1, gives the number.
(C) All nonzero numbers mod p, have an inverse.
(D) Multiplying a number by 0 gives 0.
(E) Mutliplying elements of sets A and B together is the same if $A = B$.

(A), (C), and (E)

Fermat and Exponent reducing.

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod{p}$,

$a^{p-1} \equiv 1 \pmod{p}$.

What is $2^{101} \pmod{7}$?

Wrong: $2^{101} = 2^{7*14+3} = 2^3 \pmod{7}$

Fermat: 2 is relatively prime to 7. $\implies 2^6 = 1 \pmod{7}$.

Correct: $2^{101} = 2^{6*16+5} = 2^5 = 32 = 4 \pmod{7}$.

For a prime modulus, we can reduce exponents modulo $p-1$!

Lecture in a minute.

Euclid's Alg: $gcd(x,y) = gcd(y, x \mod y)$
Fast cuz value drops by a factor of two every two recursive calls.

Extended Euclid: Find a,b where $ax+by = gcd(x,y)$.
Idea: compute a,b recursively (euclid), or iteratively.
Inverse: $ax+by = ax = gcd(x,y) \mod y$.
If $gcd(x,y) = 1$, we have $ax = 1 \mod y$
$\rightarrow a = x^{-1} mod y$.

Chinese Remainder Theorem:
If $gcd(n,m) = 1$, $x = a \pmod{n}, x = b \pmod{m}$ unique sol.

Proof: Find $u = 1 \pmod{n}$, $u = 0 \pmod{m}$,
and $v = 0 \pmod{n}$, $v = 1 \pmod{m}$.
Then: $x = au + bv = a \pmod{n}$...
$u = m(m^{-1} \pmod{n}) \pmod{n}$ works!

Fermat: Prime p, $a^{p-1} = 1 \pmod{p}$.
Proof Idea: $f(x) = a(x) \pmod{p}$: bijection on $S = \{1,\ldots,p-1\}$.
Product of elts == for range/domain: a^{p-1} factor in range.