INTEGERS

PLAN FOR TODAY

- Quotient, remainder, mod function
- Divisors, multiples
- Greatest common divisor (GCD)
- Euclid's algorithm
- Least common multiple (LCM)
- Primes and prime factorization
- Base-b expansion
- Mathematical induction

HEADS-UP FOR NEXT WEEK

Logic and proofs

Lots of basic definitions — not so fun to hear a lecture about

Please make sure to read Chapter 2 in advance, so that the lecture makes sense even if we move fast

Secs 2.1–2.4	Important; will also be covered in class
Sec 2.5	Also essential, but left for you to read
Sec 2.6	Useful and interesting, but not compulsory reading

QUOTIENT and REMAINDER

Let $d \in \mathbb{Z}^+$

For any m there exist $q \in \mathbb{Z}$ and $0 \le r < d$ s.t.

$$m = q \cdot d + r$$

$m = 12$	$d = 5$	$12 = (2) \cdot 5 + 2$
$m = 5$	$d = 12$	$5 = (0) \cdot 12 + 5$
$m = -12$	$d = 5$	$-12 = (-3) \cdot 5 + 3$
$m = -5$	$d = 12$	$-5 = (-1) \cdot 12 + 7$

Usually focus on $m \in \mathbb{Z}^+$

mod-d function: returns remainder r ($m \bmod d$)

In C, Python, F# $m \,\%\, d$

Definitions for negative numbers might vary — best to avoid in practice

$m \in \mathbb{Z}$, $d \in \mathbb{Z}^+$

$$m = q \cdot d + r \qquad 0 \le r < d$$

If $r = 0$ then
- d divides m $\quad d \mid m$
- m multiple of d

If $r \neq 0$
- d does not divide m, $\quad d \nmid m$

d divides m if $\exists\, q \in \mathbb{Z}$ s.t. $m = q \cdot d$

Ex $\quad 3 \mid 12 \qquad 5 \nmid 16$

PROPERTIES OF DIVISORS

$m, n \in \mathbb{Z}$, $d \in \mathbb{Z}^+$

(1) $m \mid m$, $1 \mid m$, $d \mid 0$

(2) If $d \mid m$ or $d \mid n$ then $d \mid mn$

(3) If $d \mid m$ and $d \mid n$, then $d \mid (sm + tn)$ $\forall s, t \in \mathbb{Z}$

(4) If $d \mid m$ and $m \mid n$, then $d \mid n$ [transitivity]

QUESTION: If $d \mid mn$, is it true that $d \mid m$ or $d \mid n$?

Proof: Just use the definition

E.g., for (3)

$m = q_1 \cdot d$ $\quad n = q_2 \cdot d$

$sm + tn = (sq_1 + tq_2)\, d$

so $d \mid (sm + tn)$ as claimed

Let $a, b, d \in \mathbb{Z}^+$

d **common divisor** of a & b if $d \mid a$ and $d \mid b$

What are the common divisors of 30 and 36

$30 = 2 \cdot 3 \cdot 5$ $\quad 36 = 2 \cdot 2 \cdot 3 \cdot 3$

Common divisors 2, 3, 6

d is **GREATEST COMMON DIVISOR** of a & b if $d \mid a$, $d \mid b$ and d is largest of all common divisors

$GCD(30, 36) = 6$

KBR THEOREM 4 (page 22)

If $d = GCD(a,b)$ then
(a) $d = sa + tb$ for some $s, t \in \mathbb{Z}$
(b) For any other common divisor c of a and b it holds that $c \mid d$

Proof

Let d^* be minimal element in set

$\{ m \mid m > 0,\ m = sa + tb,\ s,t \in \mathbb{Z} \}$

[BTW, how do we know this set has has a minimal element? LEAST NUMBER PRINCIPLE, equivalent to INDUCTION that we will talk about soon]

Suppose $c \mid a$ and $c \mid b$
Then $c \mid (sa+tb)$ by property ③
Proves (b) (assuming (a))

Write $a = qd^* + r \qquad 0 \le r < d^*$

$$\begin{aligned} r &= a - qd^* \\ &= a - q(sa + tb) \\ &= (1 - qs)a + (-qt)b \end{aligned}$$

Now we must have $r = 0$, since otherwise d^* was not minimal. Hence $d^* \mid a$

In the same way we can show that $d^* \mid b$.

The theorem follows. ▨

Theorem 4 is true, but not very constructive

To find GCD, use **EUCLIDEAN ALGORITHM**

Key ideas:

- $GCD(r, 0) = r$
- $GCD(a, b) = GCD(a \bmod b, b)$
- And $(a \bmod b) < b$

$$a = qb + r$$

$$r = a - qb$$

So any common divisor of a & b divides r

And " " b & r " " a

By property ③ again

Euclidean algorithm: $a, b \in \mathbb{Z}^+$, $a \geq b$

$GCD(a, b) = GCD(b, r_1)$ $\quad a = q_1 b + r_1$

$GCD(b, r_1) = GCD(r_1, r_2)$ $\quad b = q_2 r_1 + r_2$

$GCD(r_1, r_2) = GCD(r_2, r_3)$ $\quad r_1 = q_3 r_2 + r_3$

When $r_k = 0$, have $GCD = r_{k-1}$

$GCD(36, 30)$?

$$36 = 1 \cdot 30 + 6$$

$$30 = 5 \cdot 6 + 0$$

So $GCD = 6$

Slightly more interesting example

GCD (90, 28)

$90 = 3 \cdot 28 + 6$

$28 = 4 \cdot 6 + 4$

$6 = 1 \cdot 4 + 2$

$4 = 2 \cdot 2 + 0$ GCD = 2

Where is the linear combination

$GCD(a,b) = sa + tb$

that Thm 4 promised?

Run Euclidean algorithm backwards!

$2 = 6 - 1 \cdot 4$

$= 6 - 1 \cdot (28 - 4 \cdot 6)$

$= (-1) \cdot 28 + 5 \cdot 6$

$= (-1) \cdot 28 + 5(90 - 3 \cdot 28)$

$= (-16) \cdot 28 + 5 \cdot 90$

Let $a, b, m \in \mathbb{Z}^+$

m is a COMMON MULTIPLE of a and b if $a|m$ and $b|m$

m is the LEAST COMMON MULTIPLE of a & b, LCM(a,b) if m is the smallest of all common multiples of a & b

THEOREM $LCM(a,b) = \dfrac{ab}{GCD(a,b)}$

Ex $LCM(8,10) = 40 = \dfrac{8 \cdot 10}{2}$

Positive integer $p > 1$ is **PRIME** if only divisors are 1 and p

Ex 2, 3, 5, 7, 11, 13

Not 0, 1, −2, 4, 12

$a, b \in \mathbb{Z}^+$ are RELATIVELY PRIME if $GCD(a, b) = 1$

12 and 25 are relatively prime
12 and 15 are not

How to test if m prime?

```
PrimeTest (m)
 if  m ≤ 1 then
     return FALSE
 else
     for d := 2 up to m − 1
     |   if (m % d) = 0 then
     |       return FALSE
     return TRUE
```

floor (sqrt (m))

if m not prime, smallest factor is $\leq \sqrt{m}$

Can also loop over 2 and then odd numbers

Is this efficient?

What is the size of the input? For number-theoretic algorithms, count # bits $\lceil \log m \rceil$

Time $\sim 2^{(\log m)/2}$ exponential – NOT EFFICIENT!

But there are very efficient randomized algorithms

And in 2002, Agrawal, Kayal and Saxena presented an efficient (= polynomial-time) deterministic test

Randomized tests are better in practice

PRIME FACTORIZATION

Any $m \in \mathbb{Z}^+$ can be uniquely expressed as

$$m = p_1^{a_1} p_2^{a_2} \cdots p_k^{a_k} = \prod_{i=1}^{k} p_i^{a_i}$$

where $p_1 < p_2 < p_3 < \cdots$ primes
and all a_i ~~positive~~ non-negative integers.

d divides m if $d = \prod_{i=1}^{k} p_i^{b_i}$
for $0 \le b_i \le a_i$ for all i

Let $a = \prod_{i=1}^{k} p_i^{a_i}$

$b = \prod_{i=1}^{k} p_i^{b_i}$

Then $GCD(a,b) = \prod_{i=1}^{k} p_i^{\min(a_i, b_i)}$

$LCM(a,b) = \prod_{i=1}^{k} p_i^{\max(a_i, b_i)}$

BASE-b EXPANSION

Let $b > 1$ be a positive integer

Any $m \in \mathbb{Z}^+$ can be written as

$$m = d_k \cdot b^k + d_{k-1} \cdot b^{k-1} + \dots + d_2 b^2 + d_1 b + d_0$$

for $d_k \neq 0$ and $0 \leq d_i < b$ for all i

$$(d_k\, d_{k-1} \dots d_2\, d_1\, d_0)_b$$

is the base-b expansion

Common bases:

- Decimal $b = 10$
- Binary $b = 2$
- Octal $b = 8$
- Hexadecimal $b = 16$ digits $0, 1, \dots, 8, 9, A, B, C, D, E, F$

To find expansion, write

$$m = q_0 \cdot b + r_0 \quad \leftarrow \text{least significant digit}$$
$$q_0 = q_1 \cdot b + r_1$$
$$q_1 = q_2 \cdot b + r_2$$

Ex Write 99 in octal basis

$$99 = 12 \cdot 8 + 3$$
$$12 = 1 \cdot 8 + 4$$
$$1 = 0 \cdot 8 + 1$$

$$(143)_8$$

MATHEMATICAL INDUCTION

(1) $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$

Check for $n = 1, 2, 3, 4$ — TRUE

(2) For p a prime number

$2^p - 1$ is prime (MERSENNE PRIME NUMBER)

True for $p = 2, 3, 5, 7$.

(3) All swans are white

True for all swans we've seen so far.

(3) Is an example of empirical induction — not good enough for math

(2) fails for $p = 11$ (but works again for $p = 13, 17, 19$)

(1) Is true because you've seen it in lecture notes

But how to prove it?

"Falling Dominoes principle"

INDUCTION PRINCIPLE

If $P(n)$ is a statement for integers such that

(a) $P(n_0)$ is true — BASE CASE

(b) If for $n \geq n_0$ it holds that $P(n)$ true, then $P(n+1)$ is also true — INDUCTION STEP

Then $P(n)$ is true for all $n \geq n_0$

How to prove the induction principle?
We can't — we accept it as obviously true
An AXIOM
Follows from the LEAST NUMBER PRINCIPLE
and implies it — so they are equivalent

Prove (1) by induction

Base case ($n = 1$)

$$1^2 = \frac{1 \cdot 2 \cdot 3}{6} \quad \text{OK}$$

Induction step
Suppose

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

$$\sum_{i=1}^{n+1} i^2 = \sum_{i=1}^{n} i^2 + (n+1)^2$$

$$= \frac{n(n+1)(2n+1)}{6} + (n+1)^2 \quad \left[\text{induction hypothesis}\right]$$

$$= \frac{n(n+1)(2n+1)}{6} + \frac{6(n+1)(n+1)}{6}$$

$$= \frac{(n+1)\left(n(2n+1) + 6(n+1)\right)}{6}$$

$$= \frac{(n+1)(2n^2 + 7n + 6)}{6}$$

$$= \frac{(n+1)(n+2)(2n+3)}{6}$$

The equality follows by the induction principle

Ex Show that $5 \mid (6^n - 5n + 4) \quad \forall n \in \mathbb{Z}^+$

Base case ($n = 1$)

$6^1 - 5 \cdot 1 + 4 = 5$ OK

Induction step

Suppose $5 \mid (6^n - 5n + 4)$

$6^{n+1} - 5(n+1) + 4 =$

$= 6 \cdot 6^n - 5n - 5 + 4$

$= (6^n - 5n + 4) + (5 \cdot 6^n - 5)$

$5 \mid (6^n - 5n + 4)$ by induction hypothesis's

$5 \cdot 6^n - 5 = 5 \cdot (6^n - 1)$

so $5 \mid (6^{n+1} - 5(n+1) - 4)$

the claim follows by the induction principle

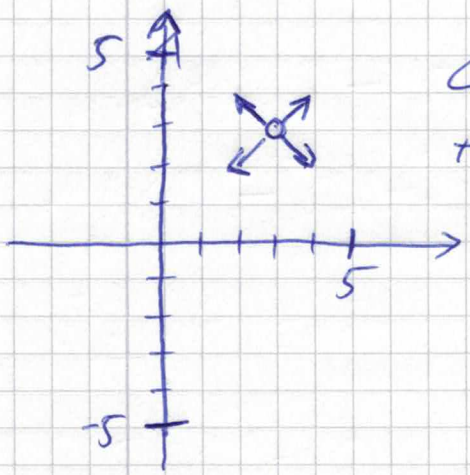
Related to **INVARIANTS**

Properties preserved throughout program or procedure

Important tool to argue correctness
Especially in for- or while-loops

Ex DIAGONALLY MOVING ROBOT IN THE PLANE

Robot starts at $(0,0)$



Can move diagonally from (x,y) to

$(x+1, y+1)$
$(x+1, y-1)$
$(x-1, y+1)$
$(x-1, y-1)$

Can the robot ever be at $(1,0)$?

Claim For any position (x,y) visited it holds that $x+y$ is an even number

Proof: Do the proof by induction

STRONG INDUCTION

If

(a) $P(n_0)$ is true, and

(b) for any $n \geq n_0$ it holds that
if $P(n_0)$, $P(n_0+1)$, ..., $P(n)$ are all true
then $P(n+1)$ true

then

$P(n)$ is true for all integers $n \geq n_0$

Actually same induction principle, but sometimes convenient to formulate it this way.

THEOREM Every positive integer ≥ 2 is a product of primes

Proof By strong induction

Base case ($n=2$)

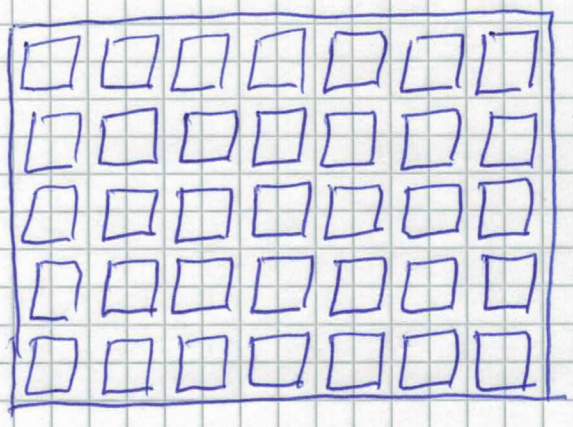
2 is a prime

Induction step

If $n+1$ is a prime, then we are done.
Otherwise, by definition $\exists\ a, b \in \mathbb{Z}^+$
such that $n+1 = a \cdot b$
By inductive hypothesis, a & b are
both product of primes. Hence,
so is $n+1$

The theorem follows by the induction principle

Given $n \times m$ chocolate bar.
How many breaks
(horizontal or vertical,
on a contiguous cake)
are needed to split the
bar into 1×1 pieces?



Use strong induction over $A = n \times m$
to prove that $A - 1$ breaks are needed

Base case (1×1 cake): 0 breaks — OK

Induction step:
Consider first break splitting the cake
into 2 pieces. Use strong induction
on each piece.

Mathematical induction can also be used for complexity analysis

Consider binary search again

```
BINARY-SEARCH (A, lo, hi, key)
    if (lo > hi)
O(1)      return -1          "BASE CASE"
    else
        mid := ⌊(lo+hi)/2⌋
        if (A[mid] == key)
            return mid
        else if (A[mid] < key)
            return BINARY-SEARCH (A, mid+1, hi, key)
        else // A[mid] > key
            return BINARY-SEARCH (A, lo, mid-1, key)
```

O(1)

Recursive call

One of these

Recursive call

Let T_n = worst-case running time for (sorted) array of size n

T_1 : Comparison
If fail, recursive call with lo > hi
terminates right away
Time: some constant K_1

T_n : Some constant K_2
plus recursive call

$$T_n \leq K_2 + T_{\lfloor n/2 \rfloor}$$

Last week, we argued that we should have

$$T_n = O(\log n)$$

Let us prove it using induction

BASE CASES

Set $K^* = K_1 + K_2$

$T_1 \leq K_1$ Ignore since $\log 1 = 0$

Enough to bound T_2 & T_3

$T_2 \leq K_2 + K_1 \leq K^* \log 2$

$T_3 \leq K_2 + K_1 \leq K^* \log 3$

INDUCTION

Suppose for all $m < n$ we have shown that

$$T_m \leq K \log m$$

for some K (which should be a constant, but we can choose it later when we see what is needed). We get

$$T_n \leq K_2 + T_{\lfloor n/2 \rfloor} \leq [\text{IH}]$$

$$\leq K_2 + K \log(\lfloor n/2 \rfloor)$$

$$\leq K_2 + K \log(n/2)$$

$$\leq K_2 + K(\log n - \log 2)$$

$$= K_2 + K \log n - K$$

$$\leq K \log n \quad \text{if} \quad K \geq K_2$$

Base cases work if $K \geq K^* = K_1 + K_2$

Set $K = K^* \Rightarrow$ induction step goes through

Formally, we have shown:

Let T_n worst-case running time for binary search on list of size n

Then there exists a constant K^* and $n_0 = 2$ so that for $n \geq n_0$

$$T_n \leq K^* \cdot \log n$$

Hence, $T_n = O(\log n)$.

Analysis of merge sort

```
MERGE-SORT(A)
    if (length(A) > 1)
        mid := ⌊(1 + length(A))/2⌋
        L := new array of size mid
        R := new array of size length(A) − mid
        for (i := 1 upto mid)
            L[i] := A[i]
        for (i := mid+1 upto length(A))
            R[i-mid] := A[i]
        MERGE-SORT(L)
        MERGE-SORT(R)
        MERGE(L, R, A)
```

O(n)

Recursive call

Recursive call

Let T_n = worst-case time complexity for array of length n

Base case ($n = 1$) is $O(1)$, so $\leq K_1$ for some K_2

In general, linear time plus 2 recursive calls for lists of half the size $n/2$, so

$$\leq K_2 n + 2 \cdot T_{n/2}$$

Actually, should be $T_{\lceil n/2 \rceil}$ if we are really careful, but we will gloss over this (which is standard)

$$T_n = \begin{cases} K_1 & \text{if } n = 1 \\ K_2 \cdot n + 2 \cdot T_{n/2} & \text{if } n > 1 \end{cases}$$

If we can prove upper bound on T_n, then get upper bound on time complexity of merge sort

T_1 $\quad$ K_1

T_2 $\quad$ $2K_2 + 2K_1$

Set $K_3 = K_2 + K_1$

Then $\boxed{T_2 \leq K_3 \cdot 2 \log 2}$

This is the BASE CASE in our inductive argument

For the INDUCTIVE STEP, assume we know for all m, $2 \leq m < n$, that

$\boxed{T_m \leq K \cdot m \log m}$ INDUCTION HYPOTHESIS

(for some K that we will pin down shortly)

Then

$$\begin{aligned} T_n &= K_2 n + 2 \cdot T_{n/2} \\ &\leq K_2 n + 2 \cdot K \cdot n/2 \log(n/2) \\ &= K_2 n + Kn(\log n - \log 2) \\ &= K_2 n + \boxed{Kn \log n} - Kn \\ &\leq Kn \log n \end{aligned}$$

if $K \geq K_2$

If we choose

$$K = K_3 \geq K_2$$

then we get (using the INDUCTION PRINCIPLE) that

> There exists a constant K and another constant n_0 $(=2)$ such that for all $n \geq n_0$ it holds that
>
> $$T_n \leq K n \log n$$

Hence, $T_n = O(n \log n)$, which is what we wanted to prove.

~~~~~~~~~~

If we wanted to be really careful, then we would have

$$T_n \leq K_2 \cdot n + T_{\lfloor n/2 \rfloor} + T_{\lceil n/2 \rceil}$$

Could still do similar calculations by using, e.g.,

$$\lfloor n/2 \rfloor \leq n/2$$

$$\lceil n/2 \rceil \leq n/2 + 1$$

$$\lceil n/2 \rceil \leq 3n/4$$

Calculations get messier, but final conclusion is the same. In computer science, we usually ignore this because we "know" these details don't matter.
~~~~~~~~~~