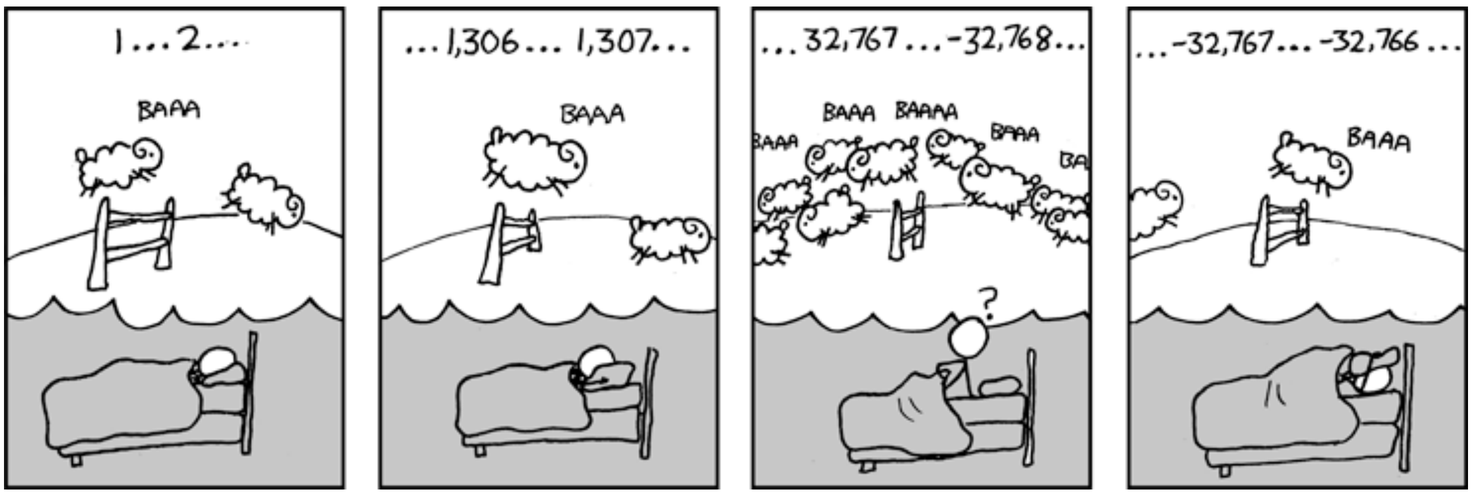
CSE 311: Foundations of Computing

Topic 5: Number Theory



Applications of Predicate Logic

- Remainder of the course will use predicate logic to prove <u>important</u> properties of <u>interesting</u> objects
  - start with math objects that are widely used in CS
  - eventually more CS-specific objects

- Encode domain knowledge in predicate definitions
- Then apply predicate logic to infer useful results

Domain of Discourse
Integers

Predicate Definitions
$\text{Even}(x) \equiv \exists y\ (x = 2 \cdot y)$
$\text{Odd}(x) \equiv \exists y\ (x = 2 \cdot y + 1)$

Mechanical vs Creative Predicate Logic

- We've done examples with "meaningless" predicates such as $\forall x\, P(x) \rightarrow \exists x\, P(x)$
  - Saw how to (often) *mechanically* solve by looking at "shape" of the goal.
  - We'll need these skills in all domains!

- When we enter "interesting" domains of discourse, we will use domain knowledge.
  - We will see how to *creatively* solve goals, especially with rules like Intro $\vee$, Intro $\exists$, Elim $\wedge$, Elim $\forall$.

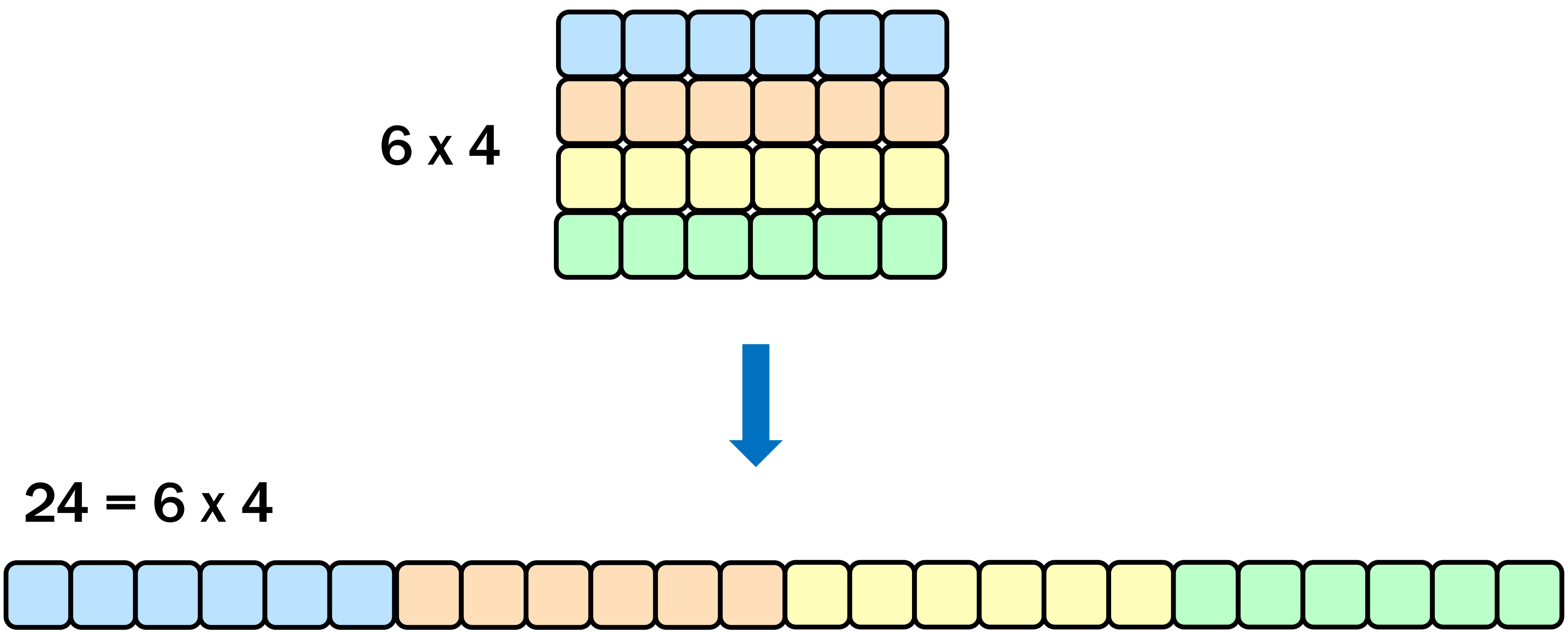
Number Theory

- Direct relevance to computing
  - everything in a computer is a number
    - colors on the screen are encoded as numbers

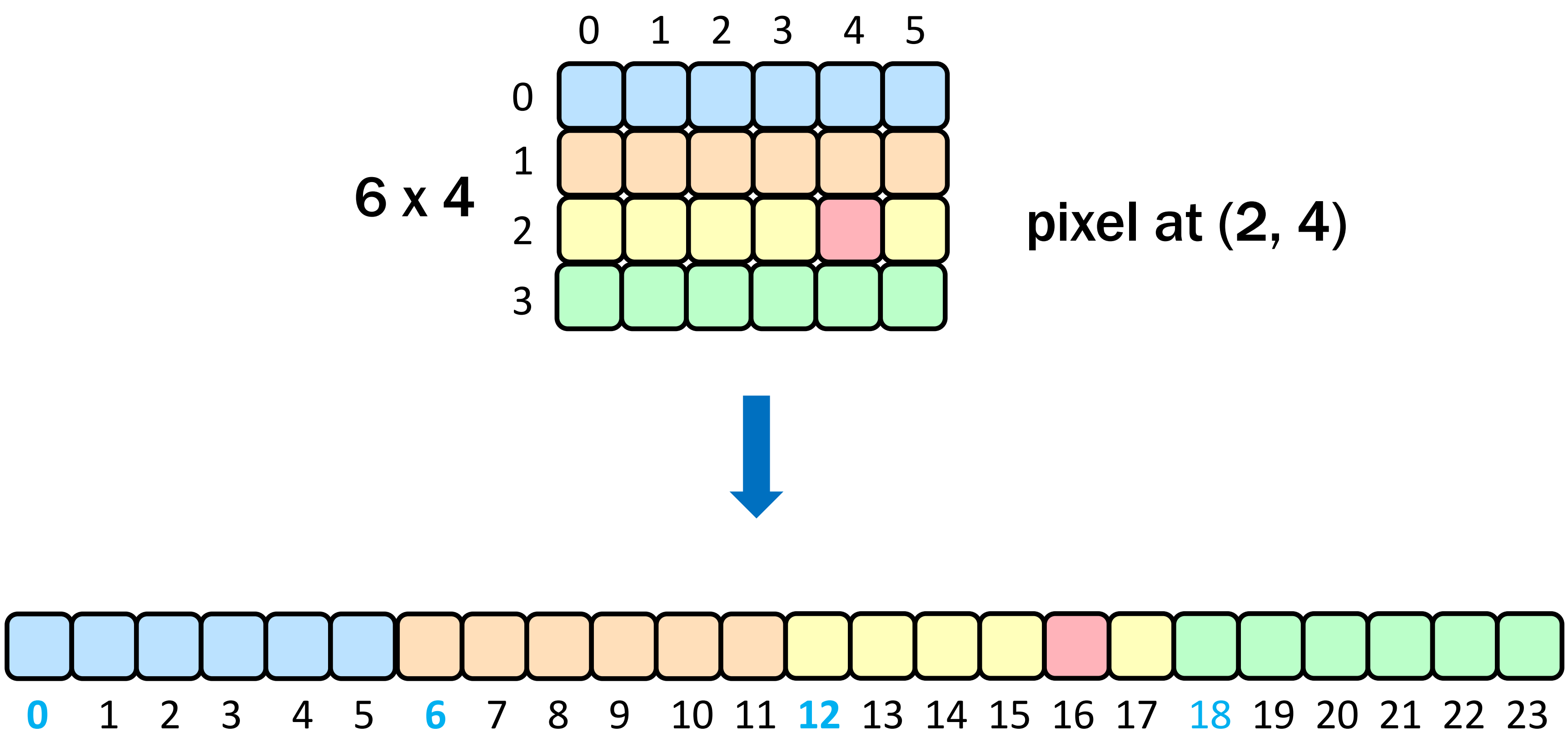
- Many significant applications in CS…

Pixels in Memory

- Memory is an array, so
  pixel positions must be mapped to array indexes



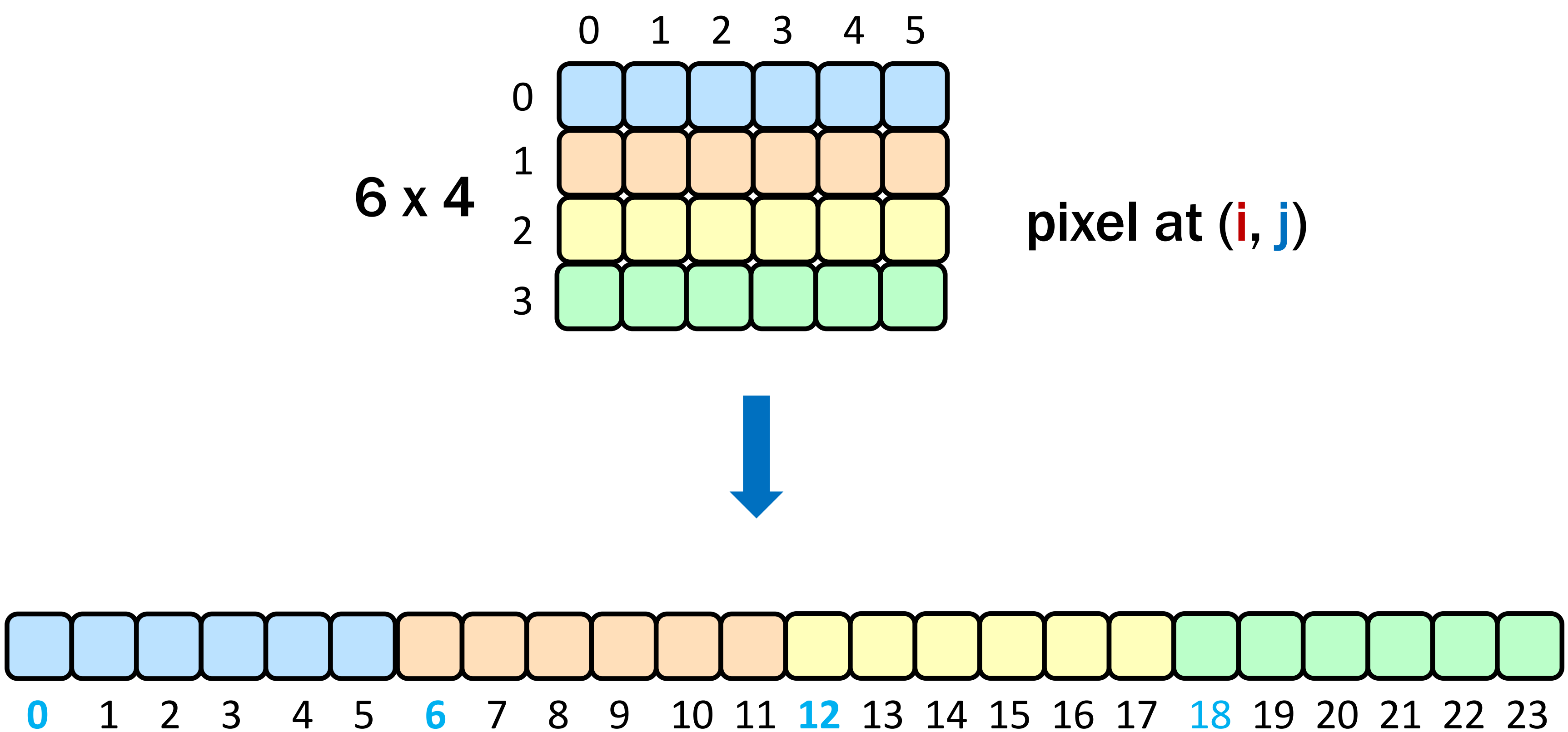
Pixels in Memory



stored at index 16 $= 12 + 4$

$= 2 \cdot 6 + 4$

Pixels in Memory



Stored at index n.

How do we calculate n from i and j?

$n = i \cdot 6 + j$

Divisibility

Domain of Discourse
Integers

Definition: "b divides a"

For a, b with $b \neq 0$:

$$b \mid a \leftrightarrow \exists q\ (a = qb)$$

Check Your Understanding. Which of the following are true?

5 | 1   25 | 5   5 | 0   3 | 2

1 | 5   5 | 25   0 | 5   2 | 3

Divisibility

Domain of Discourse
Integers

Definition: "b divides a"

For a, b with $b \neq 0$:

$$b \mid a \leftrightarrow \exists q \ (a = qb)$$

Check Your Understanding. Which of the following are true?

5 | 1
5 | 1 iff 1 = 5k

25 | 5
25 | 5 iff 5 = 25k

5 | 0
5 | 0 iff 0 = 5k

3 | 2
3 | 2 iff 2 = 3k

1 | 5
1 | 5 iff 5 = 1k

5 | 25
5 | 25 iff 25 = 5k

0 | 5
0 | 5 iff 5 = 0k

2 | 3
2 | 3 iff 3 = 2k

Recall: Elementary School Division

For a, b with $b > 0$, we can divide b into a.

If $b \mid a$, then, by definition, we have $a = qb$ for some q.
The number q is called the quotient.

Dividing both sides by b, we can write this as

$$\frac{a}{b} = q$$

(We want to stick to integers, though, so we'll write $a = qb$.)

Recall: Elementary School Division

For a, b with $b > 0$, we can divide b into a.

If $b \nmid a$, then we end up with a *remainder* r with $0 < r < b$. Now,

instead of $\frac{a}{b} = q$ we have $\frac{a}{b} = q + \frac{r}{b}$

Multiplying both sides by b gives us $a = qb + r$
(A bit nicer since it has no fractions.)

Recall: Elementary School Division

For a, b with $b > 0$, we can divide b into a.

If $b \mid a$, then we have $a = qb$ for some q.
If $b \nmid a$, then we have $a = qb + r$ for some q, r with $0 < \mathrm{r} < \mathrm{b}$.

In general, we have $a = qb + r$ for some q, r with $0 \leq r < b$, where $r = 0$ iff $b \mid a$.

Division Theorem

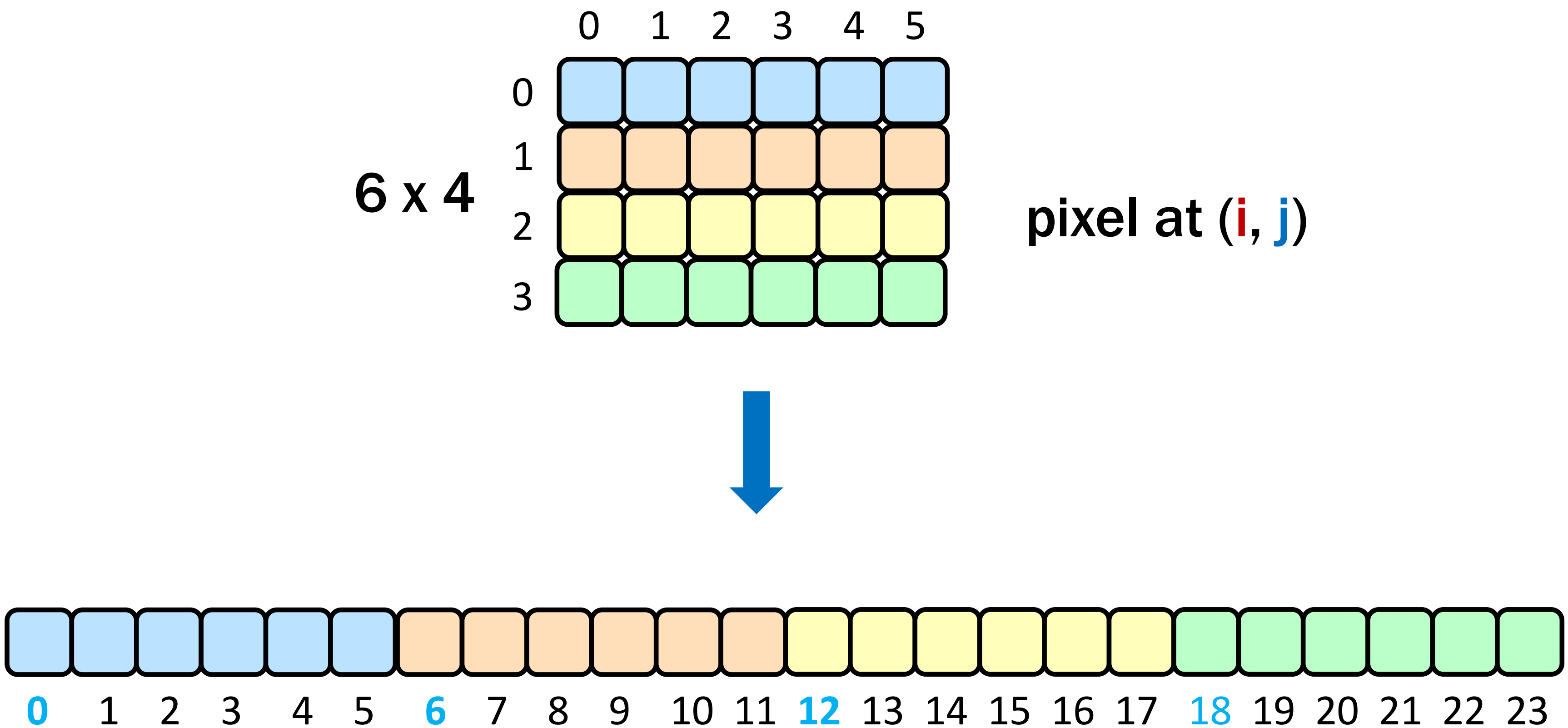
Domain of Discourse
Integers

Division Theorem

For a, b with $b > 0$
there exist *unique* integers q, r with $0 \leq r < b$
such that $a = qb + r$.

To put it another way, if we divide *b* into *a*, we get a unique quotient $q = a$ **div** b
and non-negative remainder $r = a$ **mod** b

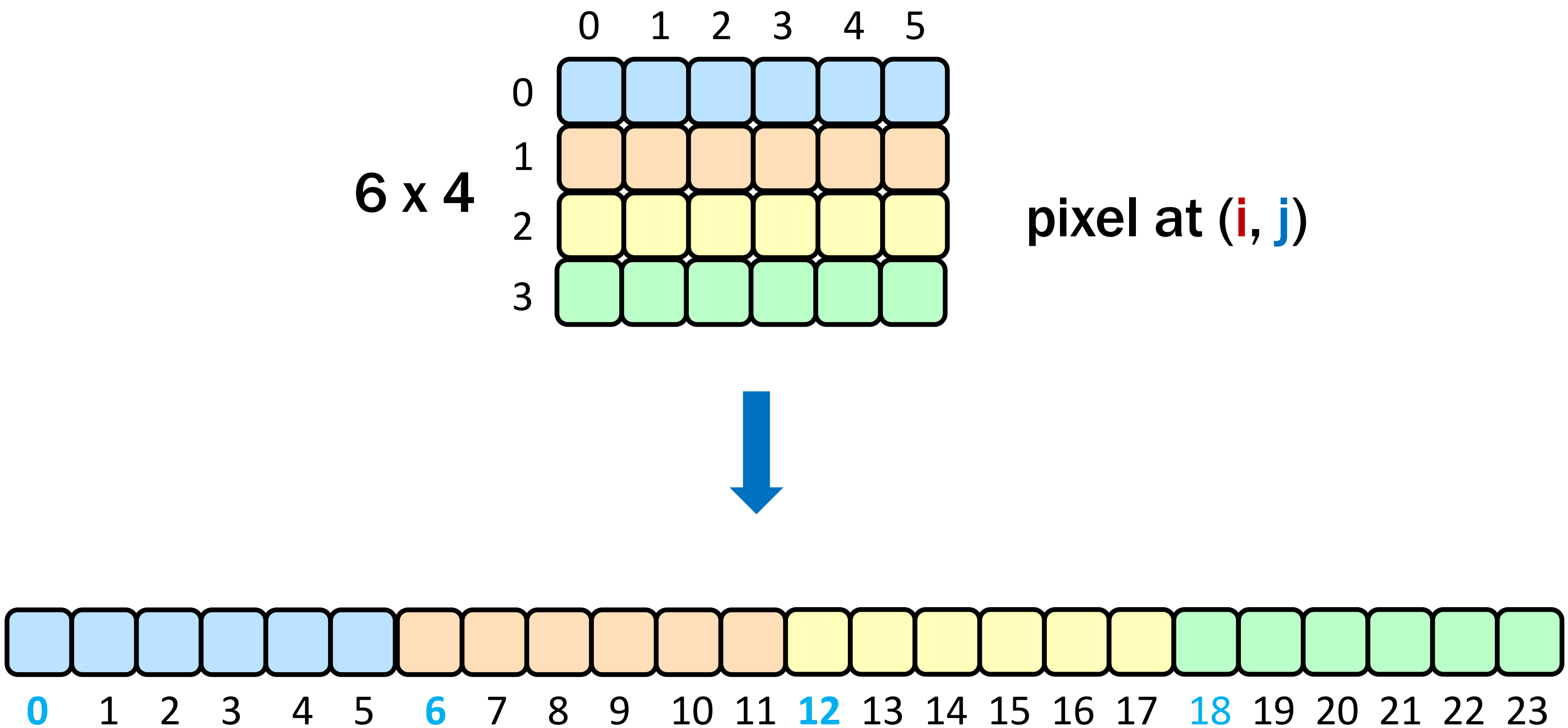
Pixels in Memory



Stored at index n.
How do we calculate n from i and j?

$$n = i \cdot 6 + j$$

Pixels in Memory



pixel at (i, j)

Stored at index n.
How do we calculate i and j from n?

i = n div 6
j = n mod 6

Number Theory

- Direct relevance to computing
  - important toolkit for programmers

- Many significant applications
  - Cryptography & Security
  - Data Structures
  - Distributed Systems

Modular Arithmetic

(and Its Applications)

Modular Arithmetic

- Arithmetic over a finite domain
- Almost all computation is over a finite domain

I’m ALIVE!

```
public class Test {
    final static int SEC_IN_YEAR = 365*24*60*60;
    public static void main(String args[]) {
        System.out.println(
            “I will be alive for at least ” +
            SEC_IN_YEAR * 101 + “ seconds.”
        );
    }
}
```

I'm ALIVE!

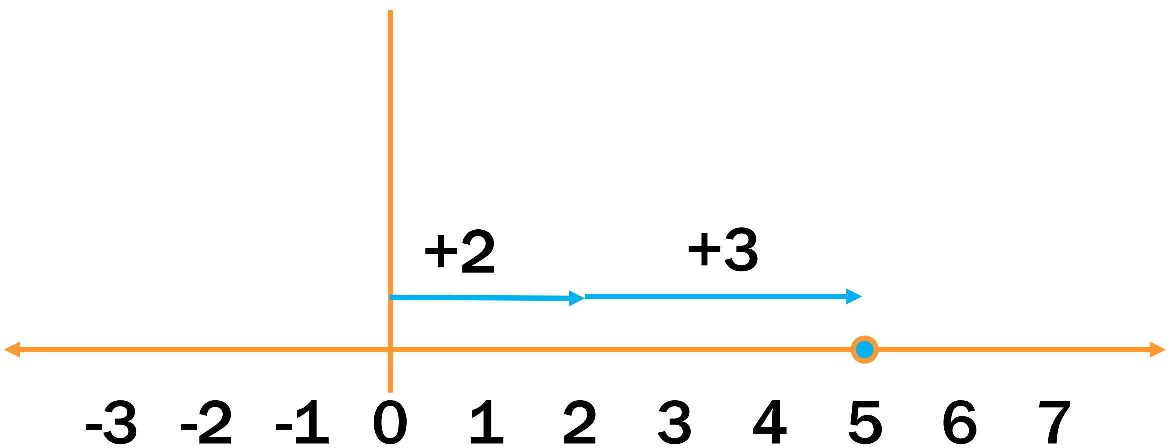
```
public class Test {
    final static int SEC_IN_YEAR = 365*24*60*60;
    public static void main(String args[]) {
        System.out.println(
            “I will be alive for at least ” +
            SEC_IN_YEAR * 101 + “ seconds.”
        );
    }
}
```

```
 ----jGRASP exec: java Test
I will be alive for at least -186619904 seconds.

 ----jGRASP: operation complete.
```

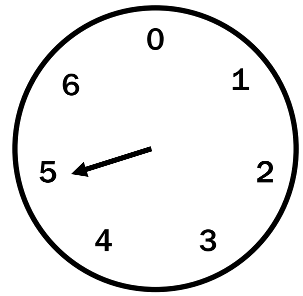
Ordinary arithmetic

$$2 + 3 = 5$$

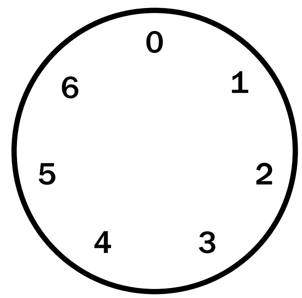


Arithmetic on a Clock

$2 + 3 = 5$



$23 = 3 \cdot 7 + 2$

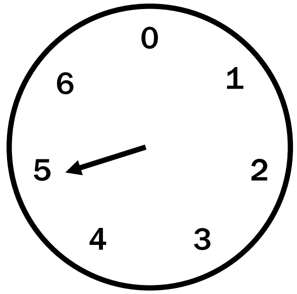


If $a = 7q + r$, then $r \;\; (= a \bmod b)$ is where you stop after taking a steps on the clock

Arithmetic, mod 7

$(a + b) \bmod 7$

$(a \times b) \bmod 7$



+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

x	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Modular Arithmetic

Domain of Discourse
Integers

Definition: "a is congruent to b modulo m"

For a, b, m with $m > 0$

$$a \equiv_m b \leftrightarrow m \mid (a - b)$$

New notion of "sameness" that will help us understand modular arithmetic

Modular Arithmetic

Domain of Discourse
Integers

Definition: "a is congruent to b modulo m"

For a, b, m with $m > 0$

$$a \equiv_m b \leftrightarrow m \mid (a - b)$$

The standard math notation is

$$a \equiv b \pmod{m}$$

A chain of equivalences is written

$$a \equiv b \equiv c \equiv d \pmod{m}$$

Many students find this confusing,
so we will use $\equiv_m$ instead.

Modular Arithmetic

Domain of Discourse
Integers

Definition: "a is congruent to b modulo m"

For a, b, m with $m > 0$

$$a \equiv_m b \;\leftrightarrow\; m \mid (a - b)$$

Check Your Understanding. What do each of these mean? When are they true?

$x \equiv_2 0$

This statement is the same as saying "x is even"; so, any x that is even (including negative even numbers) will work.

$-1 \equiv_5 19$

This statement is true. 19 - (-1) = 20 which is divisible by 5

$y \equiv_7 2$

This statement is true for y in { ..., -12, -5, 2, 9, 16, ...}. In other words, all y of the form 2+7k for k an integer.

Modular Arithmetic: A Property

Let $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{m}$ be integers with $\boldsymbol{m} > \mathbf{0}$.
Then, $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ if and only if $\boldsymbol{a} \bmod \boldsymbol{m} = \boldsymbol{b} \bmod \boldsymbol{m}$.

Modular Arithmetic: A Property

Let $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{m}$ be integers with $\boldsymbol{m} > \mathbf{0}$.
Then, $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ if and only if $\boldsymbol{a} \bmod \boldsymbol{m} = \boldsymbol{b} \bmod \boldsymbol{m}$.

Suppose that $a \bmod m = b \bmod m$.

By the division theorem, $a = mq + (a \bmod m)$ and
$b = ms + (b \bmod m)$ for some integers q,s.

Goal: show $a \equiv_m b$, i.e., $m \mid (a - b)$.

Modular Arithmetic: A Property

Let $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{m}$ be integers with $\boldsymbol{m} > \mathbf{0}$.
Then, $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ if and only if $\boldsymbol{a} \bmod \boldsymbol{m} = \boldsymbol{b} \bmod \boldsymbol{m}$.

Suppose that $a \bmod m = b \bmod m$.

By the division theorem, $a = mq + (a \bmod m)$ and
$b = ms + (b \bmod m)$ for some integers q,s.

Then, $a - b = (mq + (a \bmod m)) - (ms + (b \bmod m))$
$= m(q - s) + (a \bmod m - b \bmod m)$
$= m(q - s)$ since $a \bmod m = b \bmod m$

Goal: show $a \equiv_m b$, i.e., $m \mid (a - b)$.

Modular Arithmetic: A Property

Let $\boldsymbol{a, b, m}$ be integers with $\boldsymbol{m > 0}$.
Then, $\boldsymbol{a \equiv_m b}$ if and only if $\boldsymbol{a \bmod m = b \bmod m}$.

Suppose that $a \bmod m = b \bmod m$.

By the division theorem, $a = mq + (a \bmod m)$ and
$b = ms + (b \bmod m)$ for some integers q,s.

Then, $a - b = (mq + (a \bmod m)) - (ms + (b \bmod m))$
$= m(q - s) + (a \bmod m - b \bmod m)$
$= m(q - s)$ since $a \bmod m = b \bmod m$

Therefore, $m \mid (a - b)$ and so $a \equiv_m b$.

Modular Arithmetic: A Property

Let $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{m}$ be integers with $\boldsymbol{m} > \mathbf{0}$.
Then, $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ if and only if $\boldsymbol{a} \textbf{ mod } \boldsymbol{m} = \boldsymbol{b} \textbf{ mod } \boldsymbol{m}$.

Suppose that $a \equiv_m b$.

Then, $m \mid (a - b)$ by definition of congruence.
So, $a - b = km$ for some integer k by definition of divides.
Therefore, $a = b + km$.

Modular Arithmetic: A Property

Let $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{m}$ be integers with $\boldsymbol{m} > \mathbf{0}$.
Then, $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ if and only if $\boldsymbol{a} \textbf{ mod } \boldsymbol{m} = \boldsymbol{b} \textbf{ mod } \boldsymbol{m}$.

Suppose that $a \equiv_m b$.

Then, $m \mid (a - b)$ by definition of congruence.
So, $a - b = km$ for some integer k by definition of divides.
Therefore, $a = b + km$.

By the Division Theorem, we have $a = qm + (a \bmod m)$,
where $0 \leq (a \bmod m) < m$.

Modular Arithmetic: A Property

Let $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{m}$ be integers with $\boldsymbol{m} > \boldsymbol{0}$.
Then, $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ if and only if $\boldsymbol{a} \bmod \boldsymbol{m} = \boldsymbol{b} \bmod \boldsymbol{m}$.

Suppose that $a \equiv_m b$.

Then, $m \mid (a - b)$ by definition of congruence.
So, $a - b = km$ for some integer k by definition of divides.
Therefore, $a = b + km$.

By the Division Theorem, we have $a = qm + (a \bmod m)$,
where $0 \leq (a \bmod m) < m$.

Combining these, we have $qm + (a \bmod m) = a = b + km$
or equiv., $\mathrm{b} = qm - km + (a \bmod m) = (q - k)m + (a \bmod m)$.
By the Division Theorem, we have $b \bmod m = a \bmod m$.

The $\mathrm{mod}\ m$ function vs the $\equiv_m$ predicate

- What we have just shown
  - The $\mathrm{mod}\ m$ function maps any integer a to a remainder $a \bmod m \in \{0,1,..,m-1\}$.
  - Imagine grouping together all integers that have the same value of the $\mathrm{mod}\ m$ function

    That is, the same remainder in $\{0,1,..,m-1\}$.
  - The $\equiv_m$ predicate compares integers a, b. It is true if and only if the $\mathrm{mod}\ m$ function has the same value on a and on b.

    That is, a and b are in the same group.

Recall: Familiar Properties of "="

- If $a = b$ and $b = c$, then $a = c$.
  - i.e., if $a = b = c$, then $a = c$

- If $a = b$ and $c = d$, then $a + c = b + d$.
  - since $c = c$ is true, we can "$+\ c$" to both sides

- If $a = b$ and $c = d$, then $ac = bd$.
  - since $c = c$ is true, we can "$\times\ c$" to both sides

These facts allow us to use algebra to solve problems

Recall: Properties of "=" Used in Algebra

If $\boldsymbol{a} = \boldsymbol{b}$ and $\boldsymbol{b} = \boldsymbol{c}$, then $\boldsymbol{a} = \boldsymbol{c}$.	"Transitivity"
If $\boldsymbol{a} = \boldsymbol{b}$, then $\boldsymbol{a} + \boldsymbol{c} = \boldsymbol{b} + \boldsymbol{c}$.	"Add Equations"
If $\boldsymbol{a} = \boldsymbol{b}$, then $\boldsymbol{ac} = \boldsymbol{bc}$.	"Multiply Equations"

These are **Theorems** that
we can use in proofs

Example: **given** $5x + 4 = 2x + 25$,
prove that $3x = 21$.

Let's see how to do this in **formal** logic…

Recall: Properties of "=" Used in Algebra

If $\boldsymbol{a} = \boldsymbol{b}$ and $\boldsymbol{b} = \boldsymbol{c}$, then $\boldsymbol{a} = \boldsymbol{c}$.	"Transitivity"
If $\boldsymbol{a} = \boldsymbol{b}$, then $\boldsymbol{a} + \boldsymbol{c} = \boldsymbol{b} + \boldsymbol{c}$.	"Add Equations"
If $\boldsymbol{a} = \boldsymbol{b}$, then $\boldsymbol{ac} = \boldsymbol{bc}$.	"Multiply Equations"

1. $5x + 4 = 2x + 25$	**Given**
2. $-4 = -4$	**Algebra**
3. $5x = 2x + 21$	**Add Equations: 1, 2**
4. $-2x = -2x$	**Algebra**
5. $3x = 21$	**Add Equations: 3, 4**

Recall: Properties of "=" Used in Algebra

If $\boldsymbol{a} = \boldsymbol{b}$ and $\boldsymbol{b} = \boldsymbol{c}$, then $\boldsymbol{a} = \boldsymbol{c}$.	"Transitivity"
If $\boldsymbol{a} = \boldsymbol{b}$, then $\boldsymbol{a} + \boldsymbol{c} = \boldsymbol{b} + \boldsymbol{c}$.	"Add Equations"
If $\boldsymbol{a} = \boldsymbol{b}$, then $\boldsymbol{ac} = \boldsymbol{bc}$.	"Multiply Equations"

1. $5x + 4 = 2x + 25$ **Given**

...

5. $3x = 21$ **Transitivity**

Careful: prove $5x + 4 = 2x + 25 \Rightarrow 3x = 21$

not $3x = 21 \Rightarrow 5x + 4 = 2x + 25$

the second is a "backward" proof

Recall: Familiar Properties of "="

- If $a = b$ and $b = c$, then $a = c$.
  - i.e., if $a = b = c$, then $a = c$
- If $a = b$ and $c = d$, then $a + c = b + d$.
  - since $c = c$ is true, we can "$+ c$" to both sides
- If $a = b$ and $c = d$, then $ac = bd$.
  - since $c = c$ is true, we can "$\times c$" to both sides

Same facts apply to "≤"
with non-negative numbers

What about "$\equiv_m$"?

Modular Arithmetic: Basic Property

Let $\boldsymbol{m}$ be a positive integer.
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{b} \equiv_{\boldsymbol{m}} \boldsymbol{c}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{c}$.

Modular Arithmetic: Basic Property

Let $\boldsymbol{m}$ be a positive integer.
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{b} \equiv_{\boldsymbol{m}} \boldsymbol{c}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{c}$.

Suppose that $a \equiv_m b$ and $b \equiv_m c$.

Modular Arithmetic: Basic Property

Let $\boldsymbol{m}$ be a positive integer.
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{b} \equiv_{\boldsymbol{m}} \boldsymbol{c}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{c}$.

Suppose that $a \equiv_m b$ and $b \equiv_m c$. Then, by the previous property, we have $a \bmod m = b \bmod m$ and $b \bmod m = c \bmod m$.

Putting these together, we have $a \bmod m = c \bmod m$, which says that $a \equiv_m c$, by the previous property.

Modular Arithmetic: Addition Property

Let $\boldsymbol{m}$ be a positive integer. If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{a} + \boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{b} + \boldsymbol{d}$.

Modular Arithmetic: Addition Property

Let $\boldsymbol{m}$ be a positive integer. If $\boldsymbol{a \equiv_m b}$ and $\boldsymbol{c \equiv_m d}$, then $\boldsymbol{a + c \equiv_m b + d}$.

Suppose that $a \equiv_m b$ and $c \equiv_m d$.

Modular Arithmetic: Addition Property

Let $\boldsymbol{m}$ be a positive integer. If $\boldsymbol{a \equiv_m b}$ and $\boldsymbol{c \equiv_m d}$, then $\boldsymbol{a + c \equiv_m b + d}$.

Suppose that $a \equiv_m b$ and $c \equiv_m d$. Unrolling the definitions, we can see that $a - b = km$ and $c - d = jm$ for some $k, j \in \mathbb{Z}$.

Adding the equations together gives us
$(a + c) - (b + d) = m(k + j)$.

By the definition of congruence, we have $a + c \equiv_m b + d$.

Modular Arithmetic: Multiplication Property

Let $\boldsymbol{m}$ be a positive integer. If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bd}$.

Modular Arithmetic: Multiplication Property

Let $\boldsymbol{m}$ be a positive integer. If $\boldsymbol{a \equiv_m b}$ and $\boldsymbol{c \equiv_m d}$, then $\boldsymbol{ac \equiv_m bd}$.

Suppose that $a \equiv_m b$ and $c \equiv_m d$.

Modular Arithmetic: Multiplication Property

Let $\boldsymbol{m}$ be a positive integer. If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bd}$.

Suppose that $a \equiv_m b$ and $c \equiv_m d$. Unrolling the definitions, we can see that $a - b = km$ and $c - d = jm$ for some $k, j \in \mathbb{Z}$ or equivalently, $a = km + b$ and $c = jm + d$.

Multiplying both together gives us $ac = (km + b)(jm + d) = kjm^2 + kmd + bjm + bd$.

Modular Arithmetic: Multiplication Property

Let $\boldsymbol{m}$ be a positive integer. If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bd}$.

Suppose that $a \equiv_m b$ and $c \equiv_m d$. Unrolling the definitions, we can see that $a - b = km$ and $c - d = jm$ for some $k, j \in \mathbb{Z}$ or equivalently, $a = km + b$ and $c = jm + d$.

Multiplying both together gives us $ac = (km + b)(jm + d) = kjm^2 + kmd + bjm + bd$. Re-arranging, this becomes $ac - bd = m(kjm + kd + bj)$.

This says $ac \equiv_m bd$ by the definition of congruence.

Modular Arithmetic: Properties

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{b} \equiv_{\boldsymbol{m}} \boldsymbol{c}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{c}$.

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{a} + \boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{b} + \boldsymbol{d}$.

Corollary: If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$, then $\boldsymbol{a} + \boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{b} + \boldsymbol{c}$.

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bd}$.

Corollary: If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bc}$.

Modular Arithmetic: Properties

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{b} \equiv_{\boldsymbol{m}} \boldsymbol{c}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{c}$.

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$, then $\boldsymbol{a} + \boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{b} + \boldsymbol{c}$.

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bc}$.

“$\equiv_{m}$” allows us to solve problems in modular arithmetic, e.g.

- **add / subtract numbers from both sides of equations**
- **chains of “$\equiv_{m}$” values shows first and last are “$\equiv_{m}$”**
- **substitute “$\equiv_{m}$” values in equations (not proven yet)**

Properties of "$\equiv_m$" Used in Algebra

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{b} \equiv_{\boldsymbol{m}} \boldsymbol{c}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{c}$	"Transitivity"
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{a} + \boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{b} + \boldsymbol{d}$	"Add Equations"
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bd}$	"Multiply Equations"

These are **Theorems** that
we can use in proofs

Example: **given that** $3x \equiv_{\mathrm{m}} 7$**,**
prove that $5x + 3 \equiv_{\mathrm{m}} 2x + 10$

Properties of "$\equiv_m$" Used in Algebra

If $\boldsymbol{a} \equiv_m \boldsymbol{b}$ and $\boldsymbol{b} \equiv_m \boldsymbol{c}$, then $\boldsymbol{a} \equiv_m \boldsymbol{c}$	"Transitivity"
If $\boldsymbol{a} \equiv_m \boldsymbol{b}$ and $\boldsymbol{c} \equiv_m \boldsymbol{d}$, then $\boldsymbol{a} + \boldsymbol{c} \equiv_m \boldsymbol{b} + \boldsymbol{d}$	"Add Equations"
If $\boldsymbol{a} \equiv_m \boldsymbol{b}$ and $\boldsymbol{c} \equiv_m \boldsymbol{d}$, then $\boldsymbol{ac} \equiv_m \boldsymbol{bd}$	"Multiply Equations"

1. $3x \equiv_{\mathrm{m}} 7$ **Given**

2. $2x = 2x$ **Algebra**

3. $2x + 3x \equiv_{\mathrm{m}} 2x + 7$ **Add Equations: 2, 1 ??**

Line 2 says "=" not "$\equiv_{\mathrm{m}}$"

But "=" implies "$\equiv_{\mathrm{m}}$" !

(equality is a special case)

Properties of "$\equiv_m$" Used in Algebra

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{b} \equiv_{\boldsymbol{m}} \boldsymbol{c}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{c}$	"Transitivity"
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{a} + \boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{b} + \boldsymbol{d}$	"Add Equations"
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bd}$	"Multiply Equations"

1. $3x \equiv_{\mathrm{m}} 7$	**Given**
2. $2x = 2x$	**Algebra**
3. $2x \equiv_{\mathrm{m}} 2x$	**To Modular: 2**
4. $2x + 3x \equiv_{\mathrm{m}} 2x + 7$	**Add Equations: 3, 1**
5. $3 = 3$	**Algebra**
6. $3 \equiv_{\mathrm{m}} 3$	**To Modular**
7. $2x + 3x + 3 \equiv_{\mathrm{m}} 2x + 7 + 3$	**Add Equations: 4, 6**

Properties of "$\equiv_m$" Used in Algebra

If $a \equiv_m b$ and $b \equiv_m c$, then $a \equiv_m c$	"Transitivity"
If $a \equiv_m b$ and $c \equiv_m d$, then $a + c \equiv_m b + d$	"Add Equations"
If $a \equiv_m b$ and $c \equiv_m d$, then $ac \equiv_m bd$	"Multiply Equations"

7. $2x + 3x + 3 \equiv_m 2x + 7 + 3$	**Add Equations: 4, 6**
8. $5x + 3 = 2x + 3x + 3$	**Algebra**
9. $5x + 3 \equiv_m 2x + 3x + 3$	**To Modular: 8**
10. $2x + 7 + 3 = 2x + 10$	**Algebra**
11. $2x + 7 + 3 \equiv_m 2x + 10$	**To Modular: 10**
12. $5x + 3 \equiv_m 2x + 10$	**Transitivity: 9, 7, 11**

Good news! You'll only have to do this two times in your life...

Properties of "$\equiv_m$" Used in Algebra

If $a \equiv_m b$ and $b \equiv_m c$, then $a \equiv_m c$	"Transitivity"
If $a \equiv_m b$ and $c \equiv_m d$, then $a + c \equiv_m b + d$	"Add Equations"
If $a \equiv_m b$ and $c \equiv_m d$, then $ac \equiv_m bd$	"Multiply Equations"
If $a = b$, then $a \equiv_m b$.	"To Modular"

These are **Theorems** that
we can use in proofs

Example: **given that** $2(x-3) \equiv_{\mathrm{m}} 4$,
prove that $2x \equiv_{\mathrm{m}} 10$

Properties of "$\equiv_m$" Used in Algebra

1. $2(x-3) \equiv_{\mathrm{m}} 4$ **Given**

?. $2x \equiv_{\mathrm{m}} 10$ **??**

Properties of “$\equiv_m$” Used in Algebra

1. $2(x-3) \equiv_{\mathrm{m}} 4$	**Given**
2. $6 = 6$	**Algebra**
3. $6 \equiv_{\mathrm{m}} 6$	**To Modular: 2**
4. $2(x-3) + 6 \equiv_{\mathrm{m}} 4 + 6$	**Add Equations: 1, 3**
5. $2\mathrm{x} = 2(x-3) + 6$	**Algebra**
6. $2\mathrm{x} \equiv_{\mathrm{m}} 2(x-3) + 6$	**To Modular: 5**
7. $4 + 6 = 10$	**Algebra**
8. $4 + 6 \equiv_{\mathrm{m}} 10$	**To Modular: 7**
9. $2x \equiv_{\mathrm{m}} 10$	**Transitivity: 6, 4, 8**

Another Property of "=" Used in Algebra

If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{b} \equiv_{\boldsymbol{m}} \boldsymbol{c}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{c}$	"Transitivity"
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{a} + \boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{b} + \boldsymbol{d}$	"Add Equations"
If $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$ and $\boldsymbol{c} \equiv_{\boldsymbol{m}} \boldsymbol{d}$, then $\boldsymbol{ac} \equiv_{\boldsymbol{m}} \boldsymbol{bd}$	"Multiply Equations"
If $\boldsymbol{a} = \boldsymbol{b}$, then $\boldsymbol{a} \equiv_{\boldsymbol{m}} \boldsymbol{b}$.	"To Modular"

Can "plug in" (a.k.a. substitute)
the known value of a variable

Example: **given** $2y + 3x = 25$ **and** $x = 7$**,**
prove that $2y + 21 = 25$**.**

Substitution Follows From Other Properties

Given $2y + 3x \equiv_m 25$ **and** $x \equiv_m 7$,
show that $2y + 21 \equiv_m 25$. **(substituting** 7 **for** x**)**

Start from	$x \equiv_m 7$
Multiply both sides	$3x \equiv_m 3 \cdot 7 \qquad (= 21)$
Add to both sides	$2\mathrm{y} + 3x \equiv_m 2y + 21$
Combine $\equiv_m$**'s**	$2y + 21 \equiv_m 2\mathrm{y} + 3x \equiv_m 25$

Basic Applications of mod

- Two’s Complement
- Hashing
- Pseudo random number generation

n-bit Unsigned Integer Representation

- Represent integer x as sum of powers of 2:

99	= 64 + 32 + 2 + 1	= $2^6 + 2^5 + 2^1 + 2^0$
18	= 16 + 2	= $2^4 + 2^1$

If $b_{n-1}2^{n-1} + \cdots + b_1 2 + b_0$ with each $b_i \in \{0,1\}$
then binary representation is $b_{n-1}...b_2\ b_1\ b_0$

- For n = 8:
  99: 0110 0011
  18: 0001 0010

Easy to implement arithmetic $\mathbf{mod}\ \mathbf{2}^n$
... just throw away bits n+1 and up

$2^n \mid 2^{n+k}$ so $b_{n+k}2^{n+k} \equiv_{2^n} 0$
for $k \geq 0$

n-bit Unsigned Integer Representation

- Largest representable number is $2^n - 1$

2^n = 100…000 (n+1 bits)

$2^n - 1$ = 11…111 (n bits)

THE WALL STREET JOURNAL.

32 bits
1 = $0.0001
$429,496.7295 max

Berkshire Hathaway Inc. (BRK-A)
NYSE - Nasdaq Real Time Price. Currency in USD
436,401.00 +679.50 (+0.16%)
At close: 4:00PM EDT

Sign-Magnitude Integer Representation

n-bit signed integers

Suppose that $-2^{n-1} < x < 2^{n-1}$

First bit as the sign, $n-1$ bits for the value

99 = 64 + 32 + 2 + 1

18 = 16 + 2

For n = 8:

99: 0110 0011

-18: 1001 0010

Problem: this has both +0 and -0 (annoying)

Two's Complement Representation

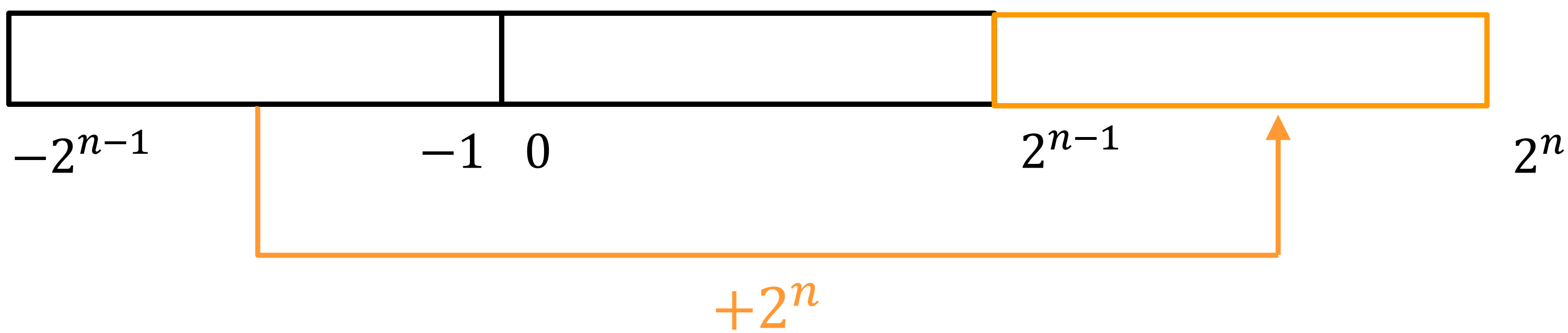
Suppose that $0 \le x < 2^{n-1}$

x is represented by the binary representation of x

Suppose that $-2^{n-1} \le x < 0$

x is represented by the binary representation of $x + 2^n$

result is in the range $2^{n-1} \le x < 2^n$



0	1	2	3	4	5	6	7	-8	-7	-6	-5	-4	-3	-2	-1
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Two's Complement Representation

Suppose that $0 \leq x < 2^{n-1}$

x is represented by the binary representation of x

Suppose that $-2^{n-1} \leq x < 0$

x is represented by the binary representation of $x + 2^n$

result is in the range $2^{n-1} \leq x < 2^n$

0	1	2	3	4	5	6	7	-8	-7	-6	-5	-4	-3	-2	-1
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

99 = 64 + 32 + 2 + 1

18 = 16 + 2

For n = 8:

99: 0110 0011

-18: 1110 1110 (-18 + 256 = 238)

Two's Complement Representation

Suppose that $0 \leq x < 2^{n-1}$
  x is represented by the binary representation of x
Suppose that $-2^{n-1} \leq x < 0$
  x is represented by the binary representation of $x + 2^n$
  result is in the range $2^{n-1} \leq x < 2^n$

0	1	2	3	4	5	6	7	-8	-7	-6	-5	-4	-3	-2	-1
0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

Key property: First bit is still the sign bit!

Key property: Twos complement representation of any number $\boldsymbol{y}$ is equivalent to $\boldsymbol{y} \bmod \boldsymbol{2^n}$ so arithmetic works $\bmod \boldsymbol{2^n}$

$$y + 2^n \equiv_{2^n} y$$

Two's Complement Representation

- For $0 < x \leq 2^{n-1}$, $-x$ is represented by the binary representation of $-x + 2^n$
  - How do we calculate –x from x?
  - E.g., what happens for "`return -x;`" in Java?

$$-x + 2^n = (2^n - 1) - \mathrm{x} + 1$$

- To compute this, flip the bits of x then add 1!
  - All 1's string is $2^n - 1$, so

    Flip the bits of x means replace x by $2^n - 1 - x$

    Then add 1 to get $-x + 2^n$

Primes

(and Their Applications)

Primality

An integer *p* greater than 1 is called *prime* if the only positive factors of *p* are 1 and *p*.

$$p > 1 \wedge \forall x\,((x \mid p) \rightarrow ((x = 1) \vee (x = p)))$$

A positive integer that is greater than 1 and is not prime is called *composite*.

$$p > 1 \wedge \exists x\,((x \mid p) \wedge (x \neq 1) \wedge (x \neq p))$$

Fundamental Theorem of Arithmetic

Every positive integer greater than 1 has a "unique" prime factorization

48 = 2 • 2 • 2 • 2 • 3
591 = 3 • 197
45,523 = 45,523
321,950 = 2 • 5 • 5 • 47 • 137
1,234,567,890 = 2 • 3 • 3 • 5 • 3,607 • 3,803

Algorithmic Problems

- **Multiplication**
  - Given primes $p_1, p_2, \ldots, p_k$, calculate their product $p_1 p_2 \ldots p_k$
- **Factoring**
  - Given an integer n, determine the prime factorization of n

Factoring

Factor the following 232 digit number [RSA768]:

123018668453011775513049495838496272077
285356959533479219732245215172640050726
365751874520219978646938995647494277406
384592519255732630345373154826850791702
612214291346167042921431160222124047927
4737794080665351419597459856902143413

12301866845301177551304949583849627207728535695953347
92197322452151726400507263657518745202199786469389956
47494277406384592519255732630345373154826850791702612
21429134616704292143116022212404792747377940806653514
19597459856902143413

=

33478071698956898786044169848212690817704794983
71376856891243138898288379387800228761471165253
1743087737814467999489

×

36746043666799590428244633799627952632279158164
34308764267603228381573966651127923337341714339
6810270092798736308917

Famous Algorithmic Problems

- Factoring
  - Given an integer n, determine the prime factorization of n
- Primality Testing
  - Given an integer n, determine if n is prime

- Factoring is hard
  - (on a classical computer)
- Primality Testing is easy

Greatest Common Divisor

(and Its Applications)

Greatest Common Divisor

GCD(a, b):

Largest integer d such that $d \mid a$ and $d \mid b$

- GCD(100, 125) =
- GCD(17, 49) =
- GCD(11, 66) =
- GCD(13, 0) =
- GCD(180, 252) =

d is GCD iff $(d \mid a) \land (d \mid b) \land \forall x\, (((x \mid a) \land (x \mid b)) \rightarrow (x \leq d))$

GCD and Factoring

$a = 2^3 \cdot 3 \cdot 5^2 \cdot 7 \cdot 11 = 46{,}200$

$b = 2 \cdot 3^2 \cdot 5^3 \cdot 7 \cdot 13 = 204{,}750$

$$GCD(a, b) = 2^{\min(3,1)} \bullet 3^{\min(1,2)} \bullet 5^{\min(2,3)} \bullet 7^{\min(1,1)} \bullet 11^{\min(1,0)} \bullet 13^{\min(0,1)}$$

Factoring is hard!

Can we compute GCD(a,b) without factoring?

Useful GCD Fact

Let *a* and *b* be positive integers.
We have gcd(*a*, *b*) = gcd(*b*, *a* mod *b*)

Proof:
We will show that every number dividing a and b also divides b and $a \bmod b$.
I.e., $d|a$ and $d|b$ iff $d|b$ and $d|(a \bmod b)$.

Hence, their set of common divisors are the same,
which means that their greatest common divisor is the same.

Useful GCD Fact

Let *a* and *b* be positive integers.
We have gcd(*a*, *b*) = gcd(*b*, *a* mod *b*)

Proof:
By definition of mod, $a = qb + (a \bmod b)$ for some integer $q = a \operatorname{div} b$.

Suppose $d|b$ and $d|(a \bmod b)$.
Then $b = md$ and $(a \bmod b) = nd$ for some integers m and n.
Therefore $a = qb + (a \bmod b) = qmd + nd = (qm + n)d$.
So $d|a$.

Suppose $d|a$ and $d|b$.
Then $a = kd$ and $b = jd$ for some integers k and j.
Therefore $(a \bmod b) = a - qb = kd - qjd = (k - qj)d$.
So, $d|(a \bmod b)$ also.

Since they have the same common divisors, $\gcd(a, b) = \gcd(b, a \bmod b)$. ∎

Another simple GCD fact

Let a be a positive integer.
We have gcd(*a*, *0*) = *a*.

Euclid's Algorithm

gcd(a, b) = gcd(b, a mod b)   gcd(a, 0) = a

```
int gcd(int a, int b){ /* Assumes: a >= b, b >= 0 */
    if (b == 0) {
        return a;
    } else {
        return gcd(b, a % b);
    }
}
```

Note: gcd(b, a) = gcd(a, b)

Euclid’s Algorithm

Repeatedly use $\gcd(a, b) = \gcd(b, a \bmod b)$ to reduce numbers until you get $\gcd(g, 0) = g$.

gcd(660,126) =

Euclid’s Algorithm

Repeatedly use $\gcd(a, b) = \gcd(b, a \bmod b)$ to reduce numbers until you get $\gcd(g, 0) = g$.

gcd(660,126) = gcd(126, 660 mod 126) = gcd(126, 30)
= gcd(30, 126 mod 30) = gcd(30, 6)
= gcd(6, 30 mod 6) = gcd(6, 0)
= 6

Bézout's theorem

If a and b are positive integers, then there exist integers $\boldsymbol{s}$ and $\boldsymbol{t}$ such that

$$\gcd(a,b) = \boldsymbol{s}a + \boldsymbol{t}b.$$

$$\forall a\, \forall b\, ((a > 0 \land b > 0) \rightarrow \exists s\, \exists t\, (\gcd(a,b) = sa + tb))$$

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Step 1 (Compute GCD & Keep Tableau Information):

a b b a mod b = r b r

$$\gcd(35, 27) = \gcd(27, 35 \bmod 27) = \gcd(27, 8)$$

a = q * b + r

$$35 = 1 * 27 + 8$$

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Step 1 (Compute GCD & Keep Tableau Information):

$$\begin{aligned}
&\overset{a \;\; b}{\gcd(35, 27)} = \overset{b \;\; a \bmod b = r}{\gcd(27, 35 \bmod 27)} = \overset{b \;\; r}{\gcd(27, 8)} \\
&= \gcd(8, 27 \bmod 8) = \gcd(8, 3) \\
&= \gcd(3, 8 \bmod 3) = \gcd(3, 2) \\
&= \gcd(2, 3 \bmod 2) = \gcd(2, 1) \\
&= \gcd(1, 2 \bmod 1) = \gcd(1, 0)
\end{aligned}$$

$$\begin{aligned}
a &= q * b + r \\
35 &= 1 * 27 + 8 \\
27 &= 3 * 8 + 3 \\
8 &= 2 * 3 + 2 \\
3 &= 1 * 2 + \textcircled{1}
\end{aligned}$$

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Step 2 (Solve the equations for r):

a = q * b + r

$35 = 1 * 27 + 8$
$27 = 3 * 8 + 3$
$8 = 2 * 3 + 2$
$3 = 1 * 2 + 1$

r = a – q * b

$8 = 35 - 1 * 27$

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Step 2 (Solve the equations for r):

a = q * b + r	r = a – q * b
$35 = 1 * 27 + 8$	$8 = 35 - 1 * 27$
$27 = 3 * 8 + 3$	$3 = 27 - 3 * 8$
$8 = 2 * 3 + 2$	$2 = 8 - 2 * 3$
$3 = 1 * 2 + 1$	$1 = 3 - 1 * 2$

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Step 3 (Backward Substitute Equations):

$8 = 35 – 1 * 27$

$3 = 27 – 3 * 8$

$2 = 8 – 2 * 3$

$1 = 3 – 1 * 2$

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Step 3 (Backward Substitute Equations):

$$8 = 35 - 1 * 27$$

$$3 = 27 - 3 * 8$$

$$2 = 8 - 2 * 3$$

$$1 = 3 - 1 * 2$$

Plug in the def of 2

$$\begin{aligned} 1 &= 3 - 1 * (8 - 2 * 3) \\ &= 3 - 8 + 2 * 3 \\ &= (-1) * 8 + 3 * 3 \end{aligned}$$

Re-arrange into 3's and 8's

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Step 3 (Backward Substitute Equations):

$$8 = 35 - 1 * 27$$

$$3 = 27 - 3 * 8$$

$$2 = 8 - 2 * 3$$

$$1 = 3 - 1 * 2$$

Plug in the def of 2

$$1 = 3 - 1 * (8 - 2 * 3)$$
$$= 3 - 8 + 2 * 3$$
$$= (-1) * 8 + 3 * 3$$

Re-arrange into 3's and 8's

Plug in the def of 3

$$= (-1) * 8 + 3 * (27 - 3 * 8)$$
$$= (-1) * 8 + 3 * 27 + (-9) * 8$$
$$= 3 * 27 + (-10) * 8$$

Re-arrange into 8's and 27's

Extended Euclidean algorithm

- Can use Euclid's Algorithm to find s, t such that

$$\gcd(a, b) = sa + tb$$

Step 3 (Backward Substitute Equations):

$$8 = 35 - 1 * 27$$

$$3 = 27 - 3 * 8$$

$$2 = 8 - 2 * 3$$

$$1 = 3 - 1 * 2$$

Plug in the def of 2

$$1 = 3 - 1 * (8 - 2 * 3)$$
$$= 3 - 8 + 2 * 3$$
$$= (-1) * 8 + 3 * 3$$

Re-arrange into 3's and 8's

Plug in the def of 3

$$= (-1) * 8 + 3 * (27 - 3 * 8)$$
$$= (-1) * 8 + 3 * 27 + (-9) * 8$$
$$= 3 * 27 + (-10) * 8$$

Re-arrange into 8's and 27's

$$= 3 * 27 + (-10) * (35 - 1 * 27)$$
$$= 3 * 27 + (-10) * 35 + 10 * 27$$
$$= 13 * 27 + (-10) * 35$$

Re-arrange into 27's and 35's

Multiplicative inverse $\mathrm{mod}\ m$

Let $0 \leq a, b < m$. Then, b is the *multiplicative inverse of* a *(modulo* m*)* iff $ab \equiv_m 1$.

x	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

mod 7

x	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	0	2	4	6	8
3	0	3	6	9	2	5	8	1	4	7
4	0	4	8	2	6	0	4	8	2	6
5	0	5	0	5	0	5	0	5	0	5
6	0	6	2	8	4	0	6	2	8	4
7	0	7	4	1	8	5	2	9	6	3
8	0	8	6	4	2	0	8	6	4	2
9	0	9	8	7	6	5	4	3	2	1

mod 10

Multiplicative inverse $\bmod m$

Suppose $\gcd(a, m) = 1$

By Bézout's Theorem, there exist integers s and t such that $sa + tm = 1$.

s is the multiplicative inverse of a (modulo m):

$$1 = sa + tm \equiv_m sa$$

So… we can compute multiplicative inverses with the extended Euclidean algorithm

These inverses let us solve modular equations…

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$

Find multiplicative inverse of 7 modulo 26

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$ **Find multiplicative inverse of 7 modulo 26**

$$\gcd(26,7) = \gcd(7,5) = \gcd(5,2) = \gcd(2,1) = 1$$

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$ **Find multiplicative inverse of 7 modulo 26**

$$\gcd(26,7) = \gcd(7,5) = \gcd(5,2) = \gcd(2,1) = 1$$

$$26 = 3 * 7 + 5$$
$$7 = 1 * 5 + 2$$
$$5 = 2 * 2 + 1$$

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$ **Find multiplicative inverse of 7 modulo 26**

$$\gcd(26,7) = \gcd(7,5) = \gcd(5,2) = \gcd(2,1) = 1$$

$$26 = 3 * 7 + 5 \qquad 5 = 26 - 3 * 7$$

$$7 = 1 * 5 + 2 \qquad 2 = 7 - 1 * 5$$

$$5 = 2 * 2 + 1 \qquad 1 = 5 - 2 * 2$$

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$ **Find multiplicative inverse of 7 modulo 26**

$$\gcd(26,7) = \gcd(7,5) = \gcd(5,2) = \gcd(2,1) = 1$$

$$26 = 3*7 + 5 \qquad 5 = 26 - 3*7$$
$$7 = 1*5 + 2 \qquad 2 = 7 - 1*5$$
$$5 = 2*2 + 1 \qquad 1 = 5 - 2*2$$

$$\begin{aligned} 1 &= 5 - 2*(7 - 1*5) \\ &= (-2)*7 + 3*5 \\ &= (-2)*7 + 3*(26 - 3*7) \\ &= (-11)*7 + 3*26 \end{aligned}$$

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$ **Find multiplicative inverse of 7 modulo 26**

$$\gcd(26,7) = \gcd(7,5) = \gcd(5,2) = \gcd(2,1) = 1$$

$$26 = 3 * 7 + 5 \qquad 5 = 26 - 3 * 7$$
$$7 = 1 * 5 + 2 \qquad 2 = 7 - 1 * 5$$
$$5 = 2 * 2 + 1 \qquad 1 = 5 - 2 * 2$$

$$\begin{aligned} 1 &= 5 - 2 * (7 - 1 * 5) \\ &= (-2) * 7 + 3 * 5 \\ &= (-2) * 7 + 3 * (26 - 3 * 7) \\ &= (-11) * 7 + 3 * 26 \end{aligned}$$

Now $(-11) \bmod 26 = 15$. ← "the" multiplicative inverse

(-11 is also "a" multiplicative inverse)

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$

Find multiplicative inverse of 7 modulo 26… it's 15.

Multiplying both sides by 15 gives

$$15 \cdot 7\mathrm{x} \equiv_{26} 15 \cdot 3$$

Simplify on both sides to get

$$\mathrm{x} \equiv_{26} 15 \cdot 7\mathrm{x} \equiv_{26} 15 \cdot 3 \equiv_{26} 19$$

So, all solutions of this congruence are numbers of the form $x = 19 + 26k$ for some $k \in \mathbb{Z}$.

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$

Conversely, suppose that $x \equiv_{26} 19$**.**

Multiplying both sides by 7 **gives**

$$7\mathrm{x} \equiv_{26} 7 \cdot 19$$

Simplify on right to get

$$7\mathrm{x} \equiv_{26} 7 \cdot 19 \equiv_{26} 3$$

So, <u>all</u> numbers of form $x = 19 + 26k$ **for any** $k \in \mathbb{Z}$ **are solutions of this equation.**

Example: Solve a Modular Equation

Solve: $7x \equiv_{26} 3$ (on HW or exams)

Step 1. **Find multiplicative inverse of 7 modulo 26**

$$1 = \ldots = (-11) * 7 \quad + \; 3 * 26$$

Since $(-11) \bmod 26 = 15$, **the inverse of** 7 **is** 15.

Step 2. **Multiply both sides and simplify**

Multiplying by 15, **we get** $x \equiv_{26} 15 \cdot 7x \equiv_{26} 15 \cdot 3 \equiv_{26} 19$.

Step 3. **State the full set of solutions**

So, the solutions are $19 + 26k$ **for any** $k \in \mathbb{Z}$

(must be of the form $a + mk$ for all $k \in \mathbb{Z}$ with $0 \leq a < m$)

Examples Not in "Standard Form"

Solve: $7x \equiv_{26} 3$

Modular equation like $\mathrm{A}x \equiv_{26} B$ **for some** A **and** B **is in "standard form".**

- solve by multiplying both sides by inverse of A

What about other equations like

$$7(x-3) \equiv_{26} 8 \text{ ?}$$

Previously saw how to formally prove this has the same solutions as equation above.

Examples Not in "Standard Form"

Solve: $7x \equiv_{26} 3$

Modular equation like $\mathrm{A}x \equiv_{26} B$ **for some** A **and** B **is in "standard form".**

- solve by multiplying both sides by inverse of A

What about other equations like

$$7(x-3) \equiv_{26} 8 \text{ ?}$$

On HW4:

- apply algorithm when in standard form (English)
- transform non-standard to standard form (formal)

Math mod a prime is especially nice

$\gcd(a, m) = 1$ if m is prime and $0 < a < m$ so can always solve these equations mod a prime.

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

x	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

mod 7

Multiplicative Inverses and Algebra

Adding to both sides is an equivalence:

$$x \equiv_m y$$

$-c$ $\quad$ $+c$

$$x + c \equiv_m y + c$$

The same is not true of multiplication...

unless we have a multiplicative inverse $cd \equiv_m 1$

$$x \equiv_m y$$

$\times d$ $\quad$ $\times c$

$$cx \equiv_m cy$$

Modular Exponentiation mod 7

x	1	2	3	4	5	6
1	1	2	3	4	5	6
2	2	4	6	1	3	5
3	3	6	2	5	1	4
4	4	1	5	2	6	3
5	5	3	1	6	4	2
6	6	5	4	3	2	1

a	a^1	a^2	a^3	a^4	a^5	a^6
1	1	1	1	1	1	1
2	2	4	1	2	4	1
3	3	2	6	4	5	1
4	4	2	1	4	2	1
5	5	4	6	2	3	1
6	6	1	6	1	6	1

Exponentiation

- **Compute** 78365^{81453}

- **Compute** $78365^{81453} \bmod 104729$

- **Output is small**
  - need to keep intermediate results small

Small Multiplications

Since $b = qm + (b \bmod m)$, we have $b \bmod m \equiv_m b$.

And since $c = tm + (c \bmod m)$, we have $c \bmod m \equiv_m c$.

Multiplying these gives $(b \bmod m)(c \bmod m) \equiv_m bc$.

By the Lemma from a few lectures ago, this tells us
$bc \bmod m = (b \bmod m)(c \bmod m) \bmod m$.

Okay to mod b and c by m before multiplying if we are planning to mod the result by m

Repeated Squaring – small and fast

Since $b \bmod m \equiv_m b$ and $c \bmod m \equiv_m c$

we have $bc \bmod m = (b \bmod m)(c \bmod m) \bmod m$

So $\quad a^2 \bmod m = (a \bmod m)^2 \bmod m$

and $\quad a^4 \bmod m = (a^2 \bmod m)^2 \bmod m$

and $\quad a^8 \bmod m = (a^4 \bmod m)^2 \bmod m$

and $\quad a^{16} \bmod m = (a^8 \bmod m)^2 \bmod m$

and $\quad a^{32} \bmod m = (a^{16} \bmod m)^2 \bmod m$

Can compute $a^k \bmod m$ for $k = 2^i$ in only i steps

What if k is not a power of 2?

Fast Exponentiation Algorithm

81453 in binary is 10011111000101101

$81453 = 2^{16} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^{9} + 2^{5} + 2^{3} + 2^{2} + 2^{0}$

$a^{81453} = a^{2^{16}} \cdot a^{2^{13}} \cdot a^{2^{12}} \cdot a^{2^{11}} \cdot a^{2^{10}} \cdot a^{2^{9}} \cdot a^{2^{5}} \cdot a^{2^{3}} \cdot a^{2^{2}} \cdot a^{2^{0}}$

a^{81453} mod m=

$(\ldots(((((a^{2^{16}}$ mod m ·
$a^{2^{13}}$ mod m) mod m ·
$a^{2^{12}}$ mod m) mod m ·
$a^{2^{11}}$ mod m) mod m ·
$a^{2^{10}}$ mod m) mod m ·
$a^{2^{9}}$ mod m) mod m ·
$a^{2^{5}}$ mod m) mod m ·
$a^{2^{3}}$ mod m) mod m ·
$a^{2^{2}}$ mod m) mod m ·
$a^{2^{0}}$ mod m) mod m

Uses only 16 + 9 = 25 multiplications

The fast exponentiation algorithm computes $a^k \bmod m$ using $\leq 2\log k$ multiplications $\bmod m$

Fast Exponentiation: $a^k \bmod m$ for all k

Another way….

$$a^{2j} \bmod m = \left(a^j \bmod m\right)^2 \bmod m$$

$$a^{2j+1} \bmod m \ = \ \big((a \bmod m) \cdot (a^{2j} \bmod m)\big) \bmod m$$

Fast Exponentiation

```
public static int FastModExp(int a, int k, int modulus) {

        if (k == 0) {
            return 1;

        } else if ((k % 2) == 0) {
            long temp = FastModExp(a,k/2,modulus);
            return (temp * temp) % modulus;

        } else {
            long temp = FastModExp(a,k-1,modulus);
            return (a * temp) % modulus;
        }
}
```

$$a^{2j} \bmod m = \left(a^{j} \bmod m\right)^{2} \bmod m$$

$$a^{2j+1} \bmod m = \left((a \bmod m) \cdot (a^{2j} \bmod m)\right) \bmod m$$

Using Fast Modular Exponentiation

- Your e-commerce web transactions use SSL (Secure Socket Layer) based on RSA encryption
- RSA
  - Vendor chooses random 512-bit or 1024-bit primes $\boldsymbol{p}, \boldsymbol{q}$ and 512/1024-bit exponent $\boldsymbol{e}$. Computes $\boldsymbol{m} = \boldsymbol{p} \cdot \boldsymbol{q}$
  - Vendor broadcasts $(\boldsymbol{m}, \boldsymbol{e})$
  - To send $\boldsymbol{a}$ to vendor, you compute $\boldsymbol{C} = \boldsymbol{a}^{\boldsymbol{e}} \bmod \boldsymbol{m}$ using *fast modular exponentiation* and send $\boldsymbol{C}$ to the vendor.
  - Using secret $\boldsymbol{p}, \boldsymbol{q}$ the vendor computes $\boldsymbol{d}$ that is the *multiplicative inverse* of $\boldsymbol{e}$ mod $(\boldsymbol{p} - \mathbf{1})(\boldsymbol{q} - \mathbf{1})$.
  - Vendor computes $\boldsymbol{C}^{\boldsymbol{d}} \bmod \boldsymbol{m}$ using *fast modular exponentiation*.
  - **Fact:** $\boldsymbol{a} = \boldsymbol{C}^{\boldsymbol{d}} \bmod \boldsymbol{m}$ for $\mathbf{0} < \boldsymbol{a} < \boldsymbol{m}$ unless $\boldsymbol{p} | \boldsymbol{a}$ or $\boldsymbol{q} | \boldsymbol{a}$