Lecture 13: Primes, GCD

For added security, after we encrypt the data stream, we send it through our Navajo code talker.

...is he just using Navajo words for "zero" and "one"?

Whoa, hey, keep your voice down!
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 $a + c = b + d$.
  - in particular, since $c = c$ is true, we can “+ $c$” to both sides

- If $a = b$ and $c = d$, then $ac = bd$.
  - in particular, since $c = c$ is true, we can “×$c$” to both sides

These are the facts that allow us to use algebra to solve problems
Let \( m \) be a positive integer.
If \( a \equiv_m b \) and \( b \equiv_m c \), then \( a \equiv_m c \).
Modular Arithmetic: Basic Property

Let $m$ be a positive integer. If $a \equiv_m b$ and $b \equiv_m c$, then $a \equiv_m c$.

Suppose that $a \equiv_m b$ and $b \equiv_m c$. 
Modular Arithmetic: Basic Property

Let $m$ be a positive integer. If $a \equiv_m b$ and $b \equiv_m c$, then $a \equiv_m c$.

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

Putting these together, we have $a \mod m = c \mod m$, which says that $a \equiv_m c$, by the previous property.
Modular Arithmetic: Addition Property

Let $m$ be a positive integer. If $a \equiv_m b$ and $c \equiv_m d$, then $a + c \equiv_m b + d$. 
Modular Arithmetic: Addition Property

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

Suppose that $a \equiv_m b$ and $c \equiv_m d$. 
Modular Arithmetic: Addition Property

Let \( m \) be a positive integer. If \( a \equiv_m b \) and \( c \equiv_m d \), then \( a + c \equiv_m b + d \).

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 \( a + c \equiv_m b + d \).
Let $m$ be a positive integer. If $a \equiv_m b$ and $c \equiv_m d$, then $ac \equiv_m bd$. 
Modular Arithmetic: Multiplication Property

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

Suppose that $a \equiv_m b$ and $c \equiv_m d$. 
Modular Arithmetic: Multiplication Property

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

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 $ac = (km + b)(jm + d) = k jm^2 + kmd + bjm + bd$. 
Let \( m \) be a positive integer. If \( a \equiv_m b \) and \( c \equiv_m d \), then \( ac \equiv_m bd \).

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 \( ac = (km + b)(jm + d) = k jm^2 + kmd + bjm + bd \). 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 $a \equiv_m b$ and $b \equiv_m c$, then $a \equiv_m c$.

If $a \equiv_m b$ and $c \equiv_m d$, then $a + c \equiv_m b + d$.

Corollary: If $a \equiv_m b$, then $a + c \equiv_m b + c$.

If $a \equiv_m b$ and $c \equiv_m d$, then $ac \equiv_m bd$.

Corollary: If $a \equiv_m b$, then $ac \equiv_m bc$. 
Modular Arithmetic: Properties

If \( a \equiv_m b \) and \( b \equiv_m c \), then \( a \equiv_m c \).

If \( a \equiv_m b \), then \( a + c \equiv_m b + c \).

If \( a \equiv_m b \), then \( ac \equiv_m bc \).

“\( \equiv_m \)” 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)
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 $3x \equiv_m 21$

Add to both sides $2y + 3x \equiv_m 2y + 21$

Combine $\equiv_m$’s $2y + 21 \equiv_m 2y + 3x \equiv_m 25$
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 $2$:

\[
\begin{align*}
99 & = 64 + 32 + 2 + 1 = 2^6 + 2^5 + 2^1 + 2^0 \\
18 & = 16 + 2 = 2^4 + 2^1
\end{align*}
\]

If $b_{n-1}2^{n-1} + \cdots + b_12 + b_0$ with each $b_i \in \{0,1\}$ then binary representation is $b_{n-1}b_{n-2}\ldots b_2 b_1 b_0$

• For $n = 8$:

<table>
<thead>
<tr>
<th>Integer</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>99</td>
<td>0110 0011</td>
</tr>
<tr>
<td>18</td>
<td>0001 0010</td>
</tr>
</tbody>
</table>

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

\[
2^n \mid 2^{n+k} \quad \text{so} \quad b_{n+k}2^{n+k} \equiv 2^n 0 \\
\text{for } k \geq 0
\]
n-bit Unsigned Integer Representation

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

\[
2^n = \underbrace{100...000}_{(n+1 \text{ bits})}
\]

\[
2^n - 1 = \underbrace{11...111}_{(n \text{ bits})}
\]
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 $0 \leq x < 2^{n-1}$
- $x$ 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 $2^{n-1} \leq x < 2^n$

---

<p>| | | | | | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>-8</td>
<td>-7</td>
<td>-6</td>
<td>-5</td>
<td>-4</td>
<td>-3</td>
<td>-2</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>0000</td>
<td>0001</td>
<td>0010</td>
<td>0011</td>
<td>0100</td>
<td>0101</td>
<td>0110</td>
<td>0111</td>
<td>1000</td>
<td>1001</td>
<td>1010</td>
<td>1011</td>
<td>1100</td>
<td>1101</td>
<td>1110</td>
<td>1111</td>
<td></td>
</tr>
</tbody>
</table>
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 $2^{n-1} \leq x < 2^n$

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>-8</th>
<th>-7</th>
<th>-6</th>
<th>-5</th>
<th>-4</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000</td>
<td>0001</td>
<td>0010</td>
<td>0011</td>
<td>0100</td>
<td>0101</td>
<td>0110</td>
<td>0111</td>
<td>1000</td>
<td>1001</td>
<td>1010</td>
<td>1011</td>
<td>1100</td>
<td>1101</td>
<td>1110</td>
<td>1111</td>
</tr>
</tbody>
</table>

$99 = 64 + 32 + 2 + 1$
$18 = 16 + 2$

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 \(2^{n-1} \leq x < 2^n\)

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>-8</th>
<th>-7</th>
<th>-6</th>
<th>-5</th>
<th>-4</th>
<th>-3</th>
<th>-2</th>
<th>-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0001</td>
<td>0010</td>
<td>0011</td>
<td>0100</td>
<td>0101</td>
<td>0110</td>
<td>0111</td>
<td>1000</td>
<td>1001</td>
<td>1010</td>
<td>1011</td>
<td>1100</td>
<td>1101</td>
<td>1110</td>
<td>1111</td>
</tr>
</tbody>
</table>

**Key property:** First bit is still the sign bit!

**Key property:** Twos complement representation of any number \(y\) is equivalent to \(y \mod 2^n\) so arithmetic works \(\mod 2^n\)

\[y + 2^n \equiv_{2^n} y\]
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?

$$-x + 2^n = (2^n - 1) - x + 1$$

• 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 $-x + 2^n$
More Number Theory
Primes and GCD
Primality

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

\[
p > 1 \land \forall x ((x \mid p) \rightarrow ((x = 1) \lor (x = p)))
\]

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

\[
p > 1 \land \exists x ((x \mid p) \land (x \neq 1) \land (x \neq p))
\]
Fundamental Theorem of Arithmetic

Every positive integer greater than 1 has a “unique” prime factorization

\[
\begin{align*}
48 &= 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \\
591 &= 3 \cdot 197 \\
45,523 &= 45,523 \\
321,950 &= 2 \cdot 5 \cdot 5 \cdot 47 \cdot 137 \\
1,234,567,890 &= 2 \cdot 3 \cdot 3 \cdot 5 \cdot 3,607 \cdot 3,803
\end{align*}
\]
Algorithmic Problems

- **Multiplication**
  - Given primes $p_1, p_2, \ldots, p_k$, calculate their product $p_1p_2 \ldots p_k$

- **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
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

GCD(a, b):

Largest integer $d$ such that $d | a$ and $d | b$

- $\text{GCD}(100, 125) = $
- $\text{GCD}(17, 49) = $
- $\text{GCD}(11, 66) = $
- $\text{GCD}(13, 0) = $
- $\text{GCD}(180, 252) = $

$d$ is GCD iff $(d | a) \land (d | b) \land \forall x (((x | a) \land (x | b)) \rightarrow (x \leq d))$
GCD and Factoring

\[ a = 2^3 \cdot 3 \cdot 5^2 \cdot 7 \cdot 11 = 46,200 \]
\[ b = 2 \cdot 3^2 \cdot 5^3 \cdot 7 \cdot 13 = 204,750 \]

\[ \text{GCD}(a, b) = 2^\min(3,1) \cdot 3^\min(1,2) \cdot 5^\min(2,3) \cdot 7^\min(1,1) \cdot 11^\min(1,0) \cdot 13^\min(0,1) \]

Factoring is hard!
Can we compute \text{GCD}(a,b) without factoring?
**Useful GCD Fact**

Let $a$ and $b$ be positive integers. We have $\text{gcd}(a,b) = \text{gcd}(b, a \mod b)$

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

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, \( a = qb + (a \mod b) \) for some integer \( q = a \div b \).

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

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

Since they have the same common divisors, \( \gcd(a, b) = \gcd(b, a \mod b) \). □
Another simple GCD fact

Let a be a positive integer. We have \( \gcd(a, 0) = a \).
Euclid’s Algorithm

\[ \text{gcd}(a, b) = \text{gcd}(b, a \mod b) \quad \text{gcd}(a, 0) = a \]

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

Note: \( \text{gcd}(b, a) = \text{gcd}(a, b) \)
Euclid’s Algorithm

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

$\gcd(660, 126) =$
Euclid’s Algorithm

Repeatedly use \( \text{gcd}(a, b) = \text{gcd}(b, a \mod b) \) to reduce numbers until you get \( \text{gcd}(g, 0) = g \).

\[
\begin{align*}
gcd(660, 126) &= gcd(126, 660 \mod 126) = gcd(126, 30) \\ &= gcd(30, 126 \mod 30) = gcd(30, 6) \\ &= gcd(6, 30 \mod 6) = gcd(6, 0) \\ &= 6
\end{align*}
\]