START
RECORDING
Mod Arithmetic
CMSC250
Divides

- We say that $a|b$ if $b = ax$ where $x \in \mathbb{Z}$
Divides

- We say that $a|b$ if $b = ax$ where $x \in \mathbb{Z}$
- Examples:
  - $2|10$
  - $5|25$
  - $5 \not{|} 7$
  - $0 \not{|} 3$
  - $8|8$
Modular Arithmetic

- We say that $a \equiv b \ (mod \ m)$ (read “$a$ is congruent to $b$ mod $m$”) means that $m \mid (a - b)$. 
Modular Arithmetic

• We say that \( a \equiv b \pmod{m} \) (read “\( a \) is congruent to \( b \) mod \( m \)”) means that \( m \mid (a - b) \).

• Examples:
  • \( 6 \equiv 2 \pmod{4} \)
  • \( 81 \equiv 0 \pmod{9} \)
  • \( 91 \equiv 0 \pmod{13} \)
  • \( 100 \equiv 2 \pmod{7} \)
Modular Arithmetic

• We say that $a \equiv b \pmod{m}$ (read “a is congruent to b mod m”) means that $m \mid (a - b)$.

• Examples:
  • $6 \equiv 2 \pmod{4}$
  • $81 \equiv 0 \pmod{9}$
  • $91 \equiv 0 \pmod{13}$
  • $100 \equiv 2 \pmod{7}$

• Convention: $0 \leq b \leq m - 1$
Modular Arithmetic

• We say that \( a \equiv b \pmod{m} \) (read “\( a \) is congruent to \( b \) mod \( m \)”) means that \( m \mid (a - b) \).

• Examples:
  • \( 6 \equiv 2 \pmod{4} \)
  • \( 81 \equiv 0 \pmod{9} \)
  • \( 91 \equiv 0 \pmod{13} \)
  • \( 100 \equiv 2 \pmod{7} \)

• Convention: \( 0 \leq b \leq m - 1 \)

• THINK: Take large number \( a \), divide by \( m \), remainder is \( b \)
Modular Arithmetic

• We say that $a \equiv b \ (mod \ m)$ (read “a is congruent to b mod m”) means that $m \mid (a - b)$.

• Examples:
  • $6 \equiv 2 \ (mod \ 4)$
  • $81 \equiv 0 \ (mod \ 9)$
  • $91 \equiv 0 \ (mod \ 13)$
  • $100 \equiv 2 \ (mod \ 7)$

• Convention: $0 \leq b \leq m - 1$

• THINK: Take large number $a$, divide by $m$, remainder is $b$

• Terminology: “Reducing $a \ mod \ m$”
\[\equiv \quad VS \quad \equiv\]

- In Logic, \(\varphi_1 \equiv \varphi_2\) mean that \(\varphi_1\) and \(\varphi_2\) have the same truth table (are logically equivalent)
\[ \equiv \quad \text{VS} \quad \equiv \]

- In Logic, \( \varphi_1 \equiv \varphi_2 \) mean that \( \varphi_1 \) and \( \varphi_2 \) have the same truth table (are logically equivalent).
- In Number Theory, \( a \equiv b \pmod{m} \), read “\( a \) is congruent to \( b \pmod{m} \)” means \( m \mid (a - b) \).
In Logic, $\varphi_1 \equiv \varphi_2$ mean that $\varphi_1$ and $\varphi_2$ have the same truth table (are logically equivalent).

In Number Theory, $a \equiv b \pmod{m}$, read “a is congruent to b mod m”) means $m \mid (a - b)$.

**THESE TWO ARE VERY DIFFERENT!!!! THEY HAVE NOTHING TO DO WITH EACH OTHER!**
Properties of congruence

1. If $a_1 \equiv b_1 \ (mod \ m)$ and $a_2 \equiv b_2 \ (mod \ m)$, then:
\[ (a_1 + a_2) \equiv (b_1 + b_2) \ (mod \ m) \]
Properties of congruence

1. If $a_1 \equiv b_1 \ (mod \ m)$ and $a_2 \equiv b_2 \ (mod \ m)$, then:
   $$(a_1 + a_2) \equiv (b_1 + b_2) \ (mod \ m)$$

Proof:
   • $a_1 \equiv b_1 \ (mod \ m) \Rightarrow m|(a_1 - b_1)$
Properties of congruence

1. If $a_1 \equiv b_1 \pmod{m}$ and $a_2 \equiv b_2 \pmod{m}$, then:

   $$(a_1 + a_2) \equiv (b_1 + b_2) \pmod{m}$$

Proof:

   • $a_1 \equiv b_1 \pmod{m} \Rightarrow m|(a_1 - b_1)$
   • $(\exists r_1 \in \mathbb{Z})[a_1 - b_1 = m \cdot r_1]$ (!)
Properties of congruence

1. If $a_1 \equiv b_1 \pmod{m}$ and $a_2 \equiv b_2 \pmod{m}$, then:
   
   $$(a_1 + a_2) \equiv (b_1 + b_2) \pmod{m}$$

Proof:

• $a_1 \equiv b_1 \pmod{m} \Rightarrow m|(a_1 - b_1)$

• $(\exists r_1 \in \mathbb{Z})[a_1 - b_1 = m \cdot r_1]$ (I)

• Similarly, $(\exists r_2 \in \mathbb{Z})[a_2 - b_2 = m \cdot r_2]$ (II)
Properties of congruence

1. If \( a_1 \equiv b_1 \pmod{m} \) and \( a_2 \equiv b_2 \pmod{m} \), then:
   \[
   (a_1 + a_2) \equiv (b_1 + b_2) \pmod{m}
   \]

Proof:

- \( a_1 \equiv b_1 \pmod{m} \Rightarrow m| (a_1 - b_1) \)
- \( (\exists r_1 \in \mathbb{Z}) [a_1 - b_1 = m \cdot r_1] \text{ (I)} \)
- Similarly, \( (\exists r_2 \in \mathbb{Z}) [a_2 - b_2 = m \cdot r_2] \text{ (II)} \)
- Therefore, by (I) and (II) we have:

\[
\begin{align*}
  a_1 - b_1 + a_2 - b_2 &= m \cdot r_1 + m \cdot r_2 \\
  (a_1 + a_2) - (b_1 + b_2) &= m \cdot (r_1 + r_2) \\
  a_1 + a_2 &\equiv (b_1 + b_2) \pmod{m}
\end{align*}
\]
Properties of congruence

2. If $a_1 \equiv b_1 \ (mod \ m)$ and $a_2 \equiv b_2 \ (mod \ m)$, then

$$a_1 \cdot a_2 \equiv b_1 \cdot b_2 \ (mod \ m)$$
Properties of congruence

Proof: Let $a_1 \equiv b_1 (\text{mod } m)$ and $a_2 \equiv b_2 (\text{mod } m)$. By definition, $jm = a_1 - b_1$ and $km = a_2 - b_2$ with $j, k \in \mathbb{Z}$. So, $jm + b_1 = a_1$ and $km + b_2 = a_2$. 
Properties of congruence

Proof: Let \( a_1 \equiv b_1 (\text{mod } m) \) and \( a_2 \equiv b_2 (\text{mod } m) \). By definition, \( jm = a_1 - b_1 \) and \( km = a_2 - b_2 \) with \( j, k \in \mathbb{Z} \). So, \( jm + b_1 = a_1 \) and \( km + b_2 = a_2 \). Then,

\[ a_1 \cdot a_2 = (jm + b_1)(km + b_2) \]
Properties of congruence

Proof: Let \( a_1 \equiv b_1 (\text{mod } m) \) and \( a_2 \equiv b_2 (\text{mod } m) \). By definition, \( jm = a_1 - b_1 \) and \( km = a_2 - b_2 \) with \( j, k \in \mathbb{Z} \). So, \( jm + b_1 = a_1 \) and \( km + b_2 = a_2 \). Then,

\[
a_1 \cdot a_2 = (jm + b_1)(km + b_2)
\]

\[
= jkm^2 + kmb_1 + jmb_2 + b_1 \cdot b_2
\]

\[
= m(jkm + kb_1 + jb_2) + b_1 \cdot b_2
\]
Properties of congruence

Proof: Let \( a_1 \equiv b_1 (mod \ m) \) and \( a_2 \equiv b_2 (mod \ m) \). By definition, \( jm = a_1 - b_1 \) and \( km = a_2 - b_2 \) with \( j, k \in \mathbb{Z} \). So, \( jm + b_1 = a_1 \) and \( km + b_2 = a_2 \). Then,

\[
a_1 \cdot a_2 = (jm + b_1)(km + b_2)
\]

\[
= jkm^2 + kmb_1 + jmb_2 + b_1 \cdot b_2
\]

\[
= m(jkm + kb_1 + jb_2) + b_1 \cdot b_2
\]

So, \( (a_1 \cdot a_2) - (b_1 \cdot b_2) = m(jkm + kb_1 + jb_2) \). Since \( jkm + kb_1 + jb_2 \in \mathbb{Z} \), \( a_1 \cdot a_2 \equiv b_1 \cdot b_2 (mod \ m) \)
Proof with modular arithmetic

• Claim: Any two integers of opposite parity sum to an odd number.
Proof with modular arithmetic

• Claim: Any two integers of opposite parity sum to an odd number.

• Proof:
  • Since $a_1, a_2$ are opposite parity. Assume that

$$a_1 \equiv 0 \pmod{2} \text{ and } a_2 \equiv 1 \pmod{2}$$
Proof with modular arithmetic

• Claim: Any two integers of opposite parity sum to an odd number.
• Proof:
  • Since $a_1, a_2$ are opposite parity. Assume that

    \[ a_1 \equiv 0 \pmod{2} \quad \text{and} \quad a_2 \equiv 1 \pmod{2} \]

  • Using the properties of modular arithmetic, we obtain:

    \[ a_1 + a_2 \equiv (0 + 1)(\pmod{2}) \equiv 1 \pmod{2} \]

• Done.
More proofs

• Similarly, you can show that $(\forall a \in \mathbb{N})[a^2 + a \equiv 0 \ (\text{mod} \ 2)]$
More proofs

• Similarly, you can show that \((\forall a \in \mathbb{N})[a^2 + a \equiv 0 \pmod{2}]\)

• Proof: We will simplify notation by assuming that “\(\equiv\)” is the same as “\(\equiv \pmod{2}\)” We have two cases:
  
  1. \(a \equiv 0\). Then, \(a^2 + a \equiv 0^2 + 0 \equiv 0\). Done.
  2. \(a \equiv 1\). Then, \(a^2 + a \equiv 1^2 + 1 \equiv 0\). Done.
Algorithms on Divisibility

1. Modular Exponentiation (Repeated Squaring)
2. Greatest Common Divisor (GCD)
Basic assumptions

• $a + b$ and $a \cdot b$ have unit cost
  • This is not true if $a, b$ are too large
First problem

How fast can we compute $a^n \mod m$ ($n, m \in \mathbb{N}$)?
First problem

How fast can we compute $a^n \mod m$ ($n, m \in \mathbb{N}$)?

1. Obviously, we can compute $a^n = a \times a \times \cdots \times a$ and mod that large number by $m$.  

\[ n \text{ times} \]
First problem

How fast can we compute \( a^n \mod m \) \((n, m \in \mathbb{N})\)?

1. Obviously, we can compute \( a^n = a \times a \times \cdots \times a \) and \( \mod \) that large number by \( m \).

• Problems
  • Arithmetic overflow in computation of \( a^n \)
  • Modding a large quantity is tough on the FPU
First problem, second approach

2. We could start computing $a \times a \times \cdots \times a$ until the product becomes larger than $m$, reduce and repeat until we’re done.
First problem, second approach

2. We could start computing $a \times a \times \cdots \times a$ until the product becomes larger than $m$, reduce and repeat until we're done.

- **Problems**
  - Arithmetic overflow in computation of $a^n$
  - Modding a large quantity is tough on the FPU
First problem, second approach

2. We could start computing $a \times a \times \cdots \times a$ until the product becomes larger than $m$, reduce and repeat until we’re done.

- **Problems**
  - Arithmetic overflow in computation of $a^n$
  - Modding a large quantity is tough on the FPU

- Additionally, we have another nice property...
First problem

• How fast can we compute $a^n \mod m$ ($n, m \in \mathbb{N}$)?

  - We always need $n$ steps
  - We can do it in roughly $\sqrt{n}$ steps
  - We can do it in roughly $\log n$ steps
  - Something Else
First problem

- How fast can we compute $a^n \mod m$ ($n, m \in \mathbb{N}$)?

- We always need $n$ steps
- We can do it in roughly $\sqrt{n}$ steps
- We can do it in roughly $\log n$ steps
- Something Else
Example

- Computing $3^{64} \mod 99$ in $\log_2 64 = 6$ steps.
Example

• Computing $3^{64} \mod 99$ in $\log_2 64 = 6$ steps.
• All $\equiv$ are $\equiv$ (mod 99).
Example

• Computing \(3^{64} \mod 99\) in \(\log_2 64 = 6\) steps.
• All \(\equiv\) are \(\equiv\) (mod 99).
  1. \(3^{2^1} \equiv 9\)
Example

- Computing $3^{64} \mod 99$ in $\log_2 64 = 6$ steps.
- All $\equiv$ are $\equiv$ (mod $99$).
  1. $3^{2^1} \equiv 9$
  2. $3^{2^2} \equiv (3^2)^2 \equiv 9^2 \equiv 81$
Example

• Computing $3^{64} \mod 99$ in $\log_2 64 = 6$ steps.
• All $\equiv$ are $\equiv \pmod{99}$.
  1. $3^{2^1} \equiv 9$
  2. $3^{2^2} \equiv (3^2)^2 \equiv 9^2 \equiv 81$
  3. $3^{2^3} \equiv (3^{2^2})^2 \equiv 81^2 \equiv 27$
Example

• Computing $3^{64} \mod 99$ in $\log_2 64 = 6$ steps.
• All $\equiv$ are $\equiv$ (mod 99).
  1. $3^{2^1} \equiv 9$
  2. $3^{2^2} \equiv (3^2)^2 \equiv 9^2 \equiv 81$
  3. $3^{2^3} \equiv (3^{2^2})^2 \equiv 81^2 \equiv 27$
  4. $3^{2^4} \equiv (3^{2^3})^2 \equiv 27^2 \equiv 36$
Example

• Computing $3^{64} \mod 99$ in $\log_2 64 = 6$ steps.
• All $\equiv$ are $\equiv$ (mod 99).
  1. $3^1 \equiv 9$
  2. $3^2 \equiv (3^2)^2 \equiv 9^2 \equiv 81$
  3. $3^3 \equiv (3^2)^2 \equiv 81^2 \equiv 27$
  4. $3^4 \equiv (3^3)^2 \equiv 27^2 \equiv 36$
  5. $3^5 \equiv (3^4)^2 \equiv 36^2 \equiv 9$
Example

- Computing $3^{64} \mod 99$ in $\log_2 64 = 6$ steps.
- All $\equiv$ are $\equiv \pmod{99}$.
  1. $3^{2^1} \equiv 9$
  2. $3^{2^2} \equiv (3^2)^2 \equiv 9^2 \equiv 81$
  3. $3^{2^3} \equiv (3^{2^2})^2 \equiv 81^2 \equiv 27$
  4. $3^{2^4} \equiv (3^{2^3})^2 \equiv 27^2 \equiv 36$
  5. $3^{2^5} \equiv (3^{2^4})^2 \equiv 36^2 \equiv 9$
  6. $3^{2^6} \equiv (9)^2 \equiv 81$
Example

- Computing $3^{64} \mod 99$ in $\log_2 64 = 6$ steps.
- All $\equiv$ are $\equiv \pmod{99}$.
  1. $3^{2^1} \equiv 9$
  2. $3^{2^2} \equiv (3^2)^2 \equiv 9^2 \equiv 81$
  3. $3^{2^3} \equiv (3^{2^2})^2 \equiv 81^2 \equiv 27$
  4. $3^{2^4} \equiv (3^{2^3})^2 \equiv 27^2 \equiv 36$
  5. $3^{2^5} \equiv (3^{2^4})^2 \equiv 36^2 \equiv 9$
  6. $3^{2^6} \equiv (9)^2 \equiv 81$
- Aha! $3^{64} = 3^{2^6} \equiv 81$
Good news, bad news

- **Good news** By using repeated squaring, we can compute $a^{2^\ell} \mod m$ quickly (roughly $\ell = \log_2 2^\ell$ steps).
Good news, bad news

• **Good news** By using repeated squaring, can compute $a^{2^\ell} \mod m$ quickly (roughly $\ell = \log_2 2^\ell$ steps)

• **Bad news** What if our exponent is **not** a power of 2?
Example

• Computing $3^{27} \mod 99$ with the same method
Example

• Computing $3^{27} \mod 99$ with the same method
• All $\equiv$ are $\equiv$ (mod 99).
  • $3^1 \equiv 3$
  • $3^2 \equiv 9$
  • $3^{2^2} \equiv (3^2)^2 \equiv 9^2 \equiv 81$
  • $3^{2^3} \equiv (3^{2^2})^2 \equiv 81^2 \equiv 27$
  • $3^{2^4} \equiv (3^{2^3})^2 \equiv 27^2 \equiv 36$
Example

• Computing $3^{27} \mod 99$ with the same method
• All $≡$ are $≡$ (mod 99).
  • $3^1 ≡ 3$
  • $3^2 ≡ 9$
  • $3^{2^2} ≡ (3^2)^2 ≡ 9^2 ≡ 81$
  • $3^{2^3} ≡ (3^{2^2})^2 ≡ 81^2 ≡ 27$
  • $3^{2^4} ≡ (3^{2^3})^2 ≡ 27^2 ≡ 36$
• $3^{27} = 3^{16} \times 3^8 \times 3^2 \times 3^1 ≡ 36 \times 27 \times 9 \times 3$
Example (contd.)

• To avoid large numbers, reduce product as you go
Example (contd.)

- To avoid large numbers, reduce product as you go

\[ 3^{27} = 3^{16} \times 3^8 \times 3^2 \times 3^1 \equiv 36 \times 27 \times 9 \times 3 \equiv (36 \times 27) \times (9 \times 3) \equiv 81 \times 27 \equiv 9 \]
Exercise

• Solve the following for $r$ please!

\[ 5^{34} \equiv r \ (mod\ 117) \]
Algorithm to compute $a^n \pmod{m}$ in $\log n$ steps

- **Step 1** Write $n = 2^{q_1} + 2^{q_2} + \cdots + 2^{q_r}$, $q_1 < q_2 < \cdots < q_r$
Algorithm to compute $a^n \pmod{m}$ in $\log n$ steps

- **Step 1** Write $n = 2^{q_1} + 2^{q_2} + \cdots + 2^{q_r}$, $q_1 < q_2 < \cdots < q_r$
- **Step 2** Note that $a^n = a^{2^{q_1}+2^{q_2}+\cdots+2^{q_r}} = a^{2^{q_1}} \times \cdots \times a^{2^{q_r}}$
Algorithm to compute \( a^n \pmod{m} \) in \( \log n \) steps

- **Step 1** Write \( n = 2^{q_1} + 2^{q_2} + \cdots + 2^{q_r}, \quad q_1 < q_2 < \cdots < q_r \)
- **Step 2** Note that \( a^n = a^{2^{q_1}+2^{q_2}+\cdots+2^{q_r}} = a^{2^{q_1}} \times \cdots \times a^{2^{q_r}} \)
- **Step 3** Use repeated squaring to compute

\[
a^{2^0}, a^{2^1}, a^{2^2}, \ldots, a^{2^{q_r}} \pmod{m}
\]

using \( a^{2^{i+1}} \equiv \left(a^{2^i}\right)^2 \pmod{m} \)
Algorithm to compute $a^n \pmod{m}$ in $\log n$ steps

- **Step 1** Write $n = 2^{q_1} + 2^{q_2} + \cdots + 2^{q_r}, \quad q_1 < q_2 < \cdots < q_r$
- **Step 2** Note that $a^n = a^{2^{q_1}+2^{q_2}+\cdots+2^{q_r}} = a^{2^{q_1}} \times \cdots \times a^{2^{q_r}}$
- **Step 3** Use repeated squaring to compute

  $$a^0, a^1, a^2, \ldots, a^{2^{q_r}} \pmod{m}$$

  using $a^{2^{i+1}} \equiv (a^{2^i})^2 \pmod{m}$

- **Step 4** Compute $a^{2^{q_1}} \times \cdots \times a^{2^{q_r}} \pmod{m}$ reducing when necessary to avoid large numbers
The key step

- The key step is Step #3. Use repeated squaring to compute

\[ a^{2^0}, a^{2^1}, a^{2^2}, \ldots, a^{2^q} \mod m \]

using

\[ a^{2^{i+1}} \equiv \left( a^{2^i} \right)^2 \pmod{m} \]
The key step

- The key step is Step #3. Use repeated squaring to compute

\[ a^{2^0}, a^{2^1}, a^{2^2}, \ldots, a^{2^{qr}} \mod m \]

using \( a^{2^{i+1}} \equiv \left( a^{2^i} \right)^2 \mod m \)

- When computing \( a^{2^{i+1}} \mod m \), already have computed \( \left( a^{2^i} \right)^2 \mod m \)
The key step

• The key step is Step #3. Use repeated squaring to compute

\[ a^{2^0}, a^{2^1}, a^{2^2}, ..., a^{2^q} \mod m \]

using \( a^{2^{i+1}} \equiv \left( a^{2^i} \right)^2 \mod m \)

• When computing \( a^{2^{i+1}} \mod m \), already have computed \( \left( a^{2^i} \right)^2 \mod m \)

• Note that all numbers are below \( m \) because we reduce \( \mod m \) every step of the way
The key step

• The key step is Step #3. Use repeated squaring to compute

\[ a^{2^0}, a^{2^1}, a^{2^2}, ..., a^{2^q} \mod m \]

using \( a^{2^{i+1}} \equiv (a^{2^i})^2 \mod m \)

• When computing \( a^{2^{i+1}} \mod m \), already have computed \( (a^{2^i})^2 \mod m \)

• Note that all numbers are below \( m \) because we reduce mod \( m \) every step of the way
  • So \( (a^{2^i})^2 \) is unit cost and anything \( \mod m \) is also unit cost!
Second problem: Greatest Common Divisor (GCD)

• If \( a, b \in \mathbb{N}^\neq 0 \), then the GCD of \( a, b \) is the \textbf{largest} non-zero integer \( n \) such that \( n | a \) and \( n | b \)
Second problem: Greatest Common Divisor (GCD)

• If $a, b \in \mathbb{N}^\neq 0$, then the GCD of $a, b$ is the largest non-zero integer $n$ such that $n \mid a$ and $n \mid b$

• What is the GCD of...
  • 10 and 15?
Second problem: Greatest Common Divisor (GCD)

• If $a, b \in \mathbb{N}^\neq 0$, then the GCD of $a, b$ is the largest non-zero integer $n$ such that $n \mid a$ and $n \mid b$

• What is the GCD of...
  • 10 and 15? 5
  • 12 and 90?
Second problem: Greatest Common Divisor (GCD)

• If $a, b \in \mathbb{N} \neq 0$, then the GCD of $a, b$ is the largest non-zero integer $n$ such that $n \mid a$ and $n \mid b$

• What is the GCD of...
  • 10 and 15? 5
  • 12 and 90? 6
  • 20 and 29?
Second problem: Greatest Common Divisor (GCD)

- If $a, b \in \mathbb{N}_{\neq 0}$, then the GCD of $a, b$ is the largest non-zero integer $n$ such that $n | a$ and $n | b$
- What is the GCD of...
  - 10 and 15? 5
  - 12 and 90? 6
  - 20 and 29? 1 (20 and 29 are called co-prime or relatively prime)
  - 153 and 181
Second problem: Greatest Common Divisor (GCD)

• If $a, b \in \mathbb{N}^{\neq 0}$, then the GCD of $a, b$ is the largest non-zero integer $n$ such that $n \mid a$ and $n \mid b$

• What is the GCD of...
  • 10 and 15? 5
  • 12 and 90? 6
  • 20 and 29? 1 (20 and 29 are called co-prime or relatively prime)
  • 153 and 181 1 (also co-prime)
Euclid’s GCD algorithm

• Recall If $a \equiv 0 \pmod{m}$ and $b \equiv 0 \pmod{m}$, then $a - b \equiv 0 \pmod{m}$
Euclid’s GCD algorithm

• Recall If $a \equiv 0 \ (mod \ m)$ and $b \equiv 0 \ (mod \ m)$, then $a - b \equiv 0 \ (mod \ m)$

• The GCD algorithm finds the greatest common divisor by executing this recursion (assume $a > b$)

$$GCD(a, b) = GCD(a, b - a)$$

Until its arguments are the same.
Greatest Common Divisor (GCD)

• **Recall** If $a \equiv 0 \pmod{m}$ and $b \equiv 0 \pmod{m}$, then $a - b \equiv 0 \pmod{m}$

• The GCD algorithm finds the **greatest** common divisor by executing this recursion *(assume $a > b$)*

$$GCD(a, b) = GCD(a, b - a)$$

Until its arguments are the same.

• **Question** If we implement this in a programming language, it **can only be done recursively**

![Yes (why)](yes.png) ![No (Why)](no.png) ![Something Else (What)](something_else.png)
Greatest Common Divisor (GCD)

• Recall If $a \equiv 0 \ (mod \ m)$ and $b \equiv 0 \ (mod \ m)$, then $a - b \equiv 0 \ (mod \ m)$

• The GCD algorithm finds the greatest common divisor by executing this recursion *(assume $a > b$)*

$$GCD(a, b) = GCD(a, b - a)$$

Until its arguments are the same.

• Question If we implement this in a programming language, it can only be done recursively

```plaintext
left = a;
right = b;
while(left != right){
    if(left > right)
        left = left - right;
    else
        right = right - left;
}
print "GCD is: " left; // Or right
```

Tail recursion
GCD example

• \( \text{GCD}(18, 100) = \)
  \( \text{GCD}(18, 100 - 18) = \text{GCD}(18, 82) = \)
  \( \text{GCD}(18, 82 - 18 = \text{GCD}(18, 64) = \)
  \( \text{GCD}(18, 64 - 18) = \text{GCD}(18, 46) = \)
  \( \text{GCD}(18, 46 - 18) = \text{GCD}(18, 28) = \)
  \( \text{GCD}(18, 28 - 18) = \text{GCD}(18, 10) = \)
  \( \text{GCD}(18 - 10, 10) = \text{GCD}(8, 10) = \)
  \( \text{GCD}(8, 10 - 8) = \text{GCD}(8, 2) = \)
  \( \text{GCD}(8 - 2, 2) = \text{GCD}(6, 2) = \)
  \( \text{GCD}(6 - 2, 2) = \text{GCD}(4, 2) = \)
  \( \text{GCD}(4 - 2, 2) = \text{GCD}(2, 2) = 2 \)
GCD example

Given integers $a, b$ with $a > b$ (without loss of generality), approximately how many steps does this algorithm take?

- $\text{GCD}(18, 100) =$
  - $\text{GCD}(18, 100 - 18) = \text{GCD}(18, 82) =$
  - $\text{GCD}(18, 82 - 18) = \text{GCD}(18, 64) =$
  - $\text{GCD}(18, 64 - 18) = \text{GCD}(18, 46) =$
  - $\text{GCD}(18, 46 - 18) = \text{GCD}(18, 28) =$
  - $\text{GCD}(18, 28 - 18) = \text{GCD}(18, 10) =$
  - $\text{GCD}(18 - 10, 10) = \text{GCD}(8, 10) =$
  - $\text{GCD}(8, 10 - 8) = \text{GCD}(8, 2) =$
  - $\text{GCD}(8 - 2, 2) = \text{GCD}(6, 2) =$
  - $\text{GCD}(6 - 2, 2) = \text{GCD}(4, 2) =$
  - $\text{GCD}(4 - 2, 2) = \text{GCD}(2, 2) = 2$
GCD example

- \( \text{GCD}(18, 100) = \)
  \[ \text{GCD}(18, 100 - 18) = \text{GCD}(18, 82) = \]
  \[ \text{GCD}(18, 82 - 18) = \text{GCD}(18, 64) = \]
  \[ \text{GCD}(18, 64 - 18) = \text{GCD}(18, 46) = \]
  \[ \text{GCD}(18, 46 - 18) = \text{GCD}(18, 28) = \]
  \[ \text{GCD}(18, 28 - 18) = \text{GCD}(18, 10) = \]
  \[ \text{GCD}(18 - 10, 10) = \text{GCD}(8, 10) = \]
  \[ \text{GCD}(8, 10 - 8) = \text{GCD}(8, 2) = \]
  \[ \text{GCD}(8 - 2, 2) = \text{GCD}(6, 2) = \]
  \[ \text{GCD}(6 - 2, 2) = \text{GCD}(4, 2) = \]
  \[ \text{GCD}(4 - 2, 2) = \text{GCD}(2, 2) = 2 \]

Given integers \(a, b\) with \(a > b\) (without loss of generality), approximately how many steps does this algorithm take?

Roughly \(\frac{a}{b}\)
Can we do better?

- \( \text{GCD}(18, 100) = \)
  \( \text{GCD}(18, 100 - 18) = \text{GCD}(18, 82) = \)
  \( \text{GCD}(18, 82 - 18) = \text{GCD}(18, 64) = \)
  \( \text{GCD}(18, 64 - 18) = \text{GCD}(18, 46) = \)
  \( \text{GCD}(18, 46 - 18) = \text{GCD}(18, 28) = \)
  \( \text{GCD}(18, 28 - 18) = \text{GCD}(18, 10) = \)
  \( \text{GCD}(18 - 10, 10) = \text{GCD}(8, 10) = \)
  \( \text{GCD}(8, 10 - 8) = \text{GCD}(8, 2) = \)
  \( \text{GCD}(8 - 2, 2) = \text{GCD}(6, 2) = \)
  \( \text{GCD}(6 - 2, 2) = \text{GCD}(4, 2) = \)
  \( \text{GCD}(4 - 2, 2) = \text{GCD}(2, 2) = 2 \)
Can we do better?

GCD(18, 100) =
GCD(18, 100 – 18) = GCD(18, 82) =
GCD(18, 82 – 18) = GCD(18, 64) =
GCD(18, 64 – 18) = GCD(18, 46) =
GCD(18, 46 – 18) = GCD(18, 28) =
GCD(18, 28 – 18) = GCD(18, 10) =
GCD(18 – 10, 10) = GCD(8, 10) =
GCD(8, 10 - 8) = GCD(8, 2) =
GCD(8 - 2, 2) = GCD(6, 2) =
GCD(6 - 2, 2) = GCD(4, 2) =
GCD(4 - 2, 2) = GCD(2, 2) = 2

GCD(18, 100 – 5 x 18) = GCD(18, 10) =
GCD(8, 10 - 8) = GCD(8, 2) =
GCD(8 - 3 x 2, 2) = GCD(2, 2) = 2

From 10 to 4 steps!
How fast is this new algorithm?

• Given non-zero integers $a, b$ with $a > b$, roughly how many steps does this new algorithm take to compute $\text{GCD}(a, b)$?

\[ \frac{a}{b^2} \quad \sqrt{a} \quad \log_2 a \quad \text{Something Else} \]
How fast is this new algorithm?

• Given non-zero integers $a, b$ with $a > b$, roughly how many steps does this new algorithm take to compute $\text{GCD}(a, b)$?

  $\frac{a}{b^2}, \sqrt{a}, \log_2 a$

• In fact, it takes $\log \phi a$, where $\phi = \frac{1+\sqrt{5}}{2}$ is the golden ratio.

• Proof by Gabriel Lamé in 1844, considered by some to be the first ever result in Algorithmic Complexity theory.
STOP
RECORDING