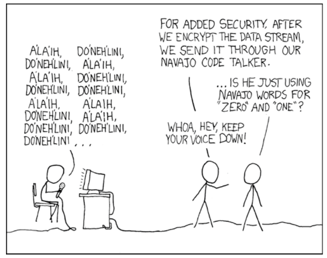
CSE 311



Foundations of Computing I

Fall 2014

Review: Division Theorem

Let a be an integer and d a positive integer. Then there are *unique* integers q and r, with $0 \le r < d$, such that $a = dq + r$.

$q = a$ **div** d $\qquad$ $r = a$ **mod** d

Review: Modular Arithmetic

Let a and b be integers, and m be a positive integer. We say a *is congruent to b modulo m* if m divides a – b. We use the notation a ≡ b (mod m) to indicate that a is congruent to b modulo m.

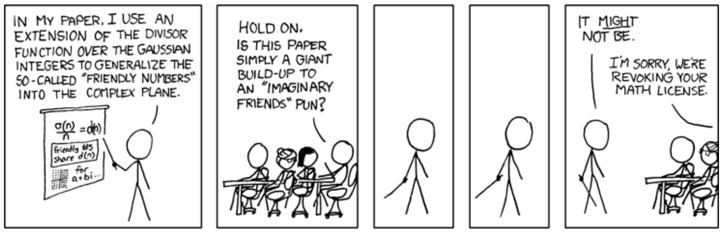
Review: Divisibility

Integers a, b, with a ≠ 0, we say that a *divides* b if there is an integer k such that b = ka. The notation a | b denotes "a divides b."

CSE 311: Foundations of Computing

Fall 2013

Lecture 11: Modular arithmetic and applications



Modular Arithmetic: A Property

Let a and b be integers, and let m be a positive integer. Then a ≡ b (mod m) if and only if a mod m = b mod m.

Modular Arithmetic: A Property

Let a and b be integers, and let m be a positive integer. Then a ≡ b (mod m) if and only if a mod m = b mod m.

Proof: Suppose that a ≡ b (mod m).

By definition: a ≡ b (mod m) implies m | (a – b) which by definition implies that a – b = km for some integer k.

Therefore a=b+km. Taking both sides modulo m we get

a mod m=(b+km) mod m = b mod m.

Suppose that a mod m = b mod m.

By the division theorem, a = mq + (a mod m) and

b = ms + (b mod m) for some integers q, s.

a – b = (mq + (a mod m)) – (ms + (b mod m))

= m(q – s) + (a mod m – b mod m)

= m(q – s) since a mod m = b mod m

Therefore m |(a-b) and so a ≡ b (mod m).

Modular Arithmetic: Another Property

Let m be a positive integer. If a ≡ b (mod m) and c ≡ d (mod m), then **a + c ≡ b + d (mod m)**

Modular Arithmetic: Another Property

Let m be a positive integer. If a ≡ b (mod m) and c ≡ d (mod m), then **a + c ≡ b + d (mod m)**

Suppose a ≡ b (mod m) and c ≡ d (mod m). Unrolling definitions gives us some integer k such that
a – b = km, and some integer j such that c – d = jm.

Adding the equations together gives us
(a + c) – (b + d) = m(k + j). Now, re-applying the definition of mod gives us a + c ≡ b + d (mod m).

Modular Arithmetic: Another-nother Property

Let m be a positive integer. If a ≡ b (mod m) and c ≡ d (mod m), then **ac ≡ bd (mod m)**

Modular Arithmetic: Another-nother Property

Let m be a positive integer. If a ≡ b (mod m) and c ≡ d (mod m), then **ac ≡ bd (mod m)**

Suppose a ≡ b (mod m) and c ≡ d (mod m). Unrolling definitions gives us some integer k such that
a – b = km, and some integer j such that c – d = jm.

Then, a = km + b and c = jm + d. Multiplying both together gives us ac = (km + b)(jm + d) = kjm^2 + kmd + jmb + bd.

Re-arranging gives us ac – bd = m(kjm + kd + jb). Using the definition of mod gives us ac ≡ bd (mod m).

Example

Let n be an integer.
Prove that $n^2 \equiv 0 \pmod 4$ or $n^2 \equiv 1 \pmod 4$

Example

Let n be an integer.
Prove that $n^2 \equiv 0 \pmod 4$ or $n^2 \equiv 1 \pmod 4$

Let's start by looking at a small example:

$0^2 = 0 \equiv 0 \pmod 4$
$1^2 = 1 \equiv 1 \pmod 4$
$2^2 = 4 \equiv 0 \pmod 4$
$3^2 = 9 \equiv 1 \pmod 4$
$4^2 = 16 \equiv 0 \pmod 4$

It looks like

$n \equiv 0 \pmod 2 \rightarrow n^2 \equiv 0 \pmod 4$, and
$n \equiv 1 \pmod 2 \rightarrow n^2 \equiv 1 \pmod 4$.

Example

Let n be an integer.
Prove that $n^2 \equiv 0 \pmod 4$ or $n^2 \equiv 1 \pmod 4$

Let's start by looking at a small example:

$0^2 = 0 \equiv 0 \pmod 4$
$1^2 = 1 \equiv 1 \pmod 4$
$2^2 = 4 \equiv 0 \pmod 4$
$3^2 = 9 \equiv 1 \pmod 4$
$4^2 = 16 \equiv 0 \pmod 4$

It looks like

$n \equiv 0 \pmod 2 \rightarrow n^2 \equiv 0 \pmod 4$, and
$n \equiv 1 \pmod 2 \rightarrow n^2 \equiv 1 \pmod 4$.

Case 1 (n is even):
Suppose $n \equiv 0 \pmod 2$.
Then, $n = 2k$ for some integer k.
So, $n^2 = (2k)^2 = 4k^2$. So, by definition of congruence, $n^2 \equiv 0 \pmod 4$.

Case 2 (n is odd):
Suppose $n \equiv 1 \pmod 2$.
Then, $n = 2k + 1$ for some integer k.
So, $n^2 = (2k + 1)^2 = 4k^2 + 4k + 1 = 4(k^2 + k) + 1$. So, by definition of congruence, $n^2 \equiv 1 \pmod 4$.

n-bit Unsigned Integer Representation

- Represent integer x as sum of powers of 2:
  If $x = \sum_{i=0}^{n-1} b_i 2^i$ where each $b_i \in \{0,1\}$
  then representation is $b_{n-1} \ldots b_2\, b_1\, b_0$

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

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

Sign-Magnitude Integer Representation

n-bit signed integers
Suppose $-2^{n-1} < x < 2^{n-1}$
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

Any problems with this representation?

Two's Complement Representation

n bit signed integers, first bit will still be the sign bit

Suppose $0 \leq x < 2^{n-1}$,
x is represented by the binary representation of x
Suppose $0 \leq x \leq 2^{n-1}$,
$-x$ is represented by the binary representation of $2^n - x$

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

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

For n = 8:
99: 0110 0011
-18: 1110 1110

Sign-Magnitude vs. Two's Complement

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

Sign-Magnitude

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

Two's complement

Two's Complement Representation

- For $0 < x \leq 2^{n-1}$, $-x$ is represented by the binary representation of $2^n - x$
- 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 $\equiv$ replace x by $2^n - 1 - x$

Basic Applications of mod

- **Hashing**
- **Pseudo random number generation**
- **Simple cipher**

Hashing

Scenario:

Map a small number of data values from a large domain $\{0, 1, \dots, M-1\}$ ...

...into a small set of locations $\{0,1, \dots, n-1\}$ so one can quickly check if some value is present

- $\text{hash}(x) = x \bmod p$ **for** p **a prime close to** n
  - **or** $\text{hash}(x) = (ax + b) \bmod p$
- **Depends on all of the bits of the data**
  - **helps avoid collisions due to similar values**
  - **need to manage them if they occur**

Pseudo-Random Number Generation

Linear Congruential method

$$x_{n+1} = (a\, x_n + c) \bmod m$$

Choose random x_0, a, c, m and produce a long sequence of x_n's

Simple Ciphers

- **Caesar cipher,** A = 1, B = 2, . . .
  - HELLO WORLD
- **Shift cipher**
  - f(p) = (p + k) mod 26
  - f^{-1}(p) = (p – k) mod 26
- **More general**
  - f(p) = (ap + b) mod 26

modular exponentiation mod 7

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

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

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						
2						
3						
4						
5						
6						

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