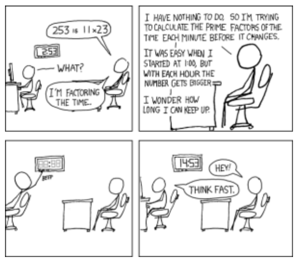
cse 311: foundations of computing

Spring 2015
Lecture 12: Primes, GCD, applications



casting out 3s

Theorem: A positive integer n is divisible by 3 if and only if the sum of its decimal digits is divisible by 3.

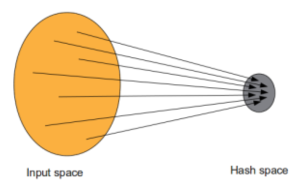
basic applications of mod

- 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.



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:

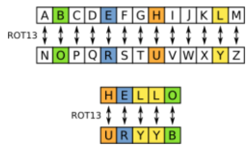
$$x_{n+1} = (a\, x_n + c) \bmod m$$

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

[good for some applications, really bad for many others]

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

exponentiation

- Compute 78365^{81453}
- Compute 78365^{81453} mod 104729
- Output is small
  - need to keep intermediate results small

repeated squaring – small and fast

Since a mod m ≡ a (mod m) for any a

we have a^2 mod m = $(a \text{ mod } m)^2$ mod m
and a^4 mod m = $(a^2 \text{ mod } m)^2$ mod m
and a^8 mod m = $(a^4 \text{ mod } m)^2$ mod m
and a^{16} mod m = $(a^8 \text{ mod } m)^2$ mod m
and a^{32} mod m = $(a^{16} \text{ mod } m)^2$ mod m

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

fast exponentiaion

```
public static long FastModExp(long base, long exponent, long modulus) {
      long result = 1;
      base = base % modulus;

      while (exponent > 0) {
          if ((exponent % 2) == 1) {
              result = (result * base) % modulus;
               exponent -= 1;
          }
          /* Note that exponent is definitely divisible by 2 here. */
          exponent /= 2;
          base = (base * base) % modulus;
          /* The last iteration of the loop will always be exponent = 1 */
          /* so, result will always be correct. */
      }
      return result;
  }
```

$b^e \bmod m = (b^2)^{e/2} \bmod m$, when e is even)
$b^e \bmod m = (b*(b^{e-1} \bmod m) \bmod m)) \bmod m$

program trace

Let M = 104729

$78365^{81453} \bmod M$
$= ((78365 \bmod M) * (78365^{81452} \bmod M)) \bmod M$
$= (78365 * ((78365^2 \bmod M)^{81452/2} \bmod M)) \bmod M$
$= (78365 * ((78852)^{40726} \bmod M)) \bmod M$
$= (78365 * ((78852^2 \bmod M)^{20363} \bmod M)) \bmod M$
$= (78365 * (86632^{20363} \bmod M)) \bmod M$
$= (78365 * ((86632 \bmod M)* (86632^{20362} \bmod M)) \bmod M$
= ...
= 45235

fast exponentiation algorithm

Another way:

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

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

$a^{81453} \bmod m =$
$(\ldots(((((a^{2^{16}} \bmod m \cdot$
$a^{2^{13}} \bmod m) \bmod m \cdot$
$a^{2^{12}} \bmod m) \bmod m \cdot$
$a^{2^{11}} \bmod m) \bmod m \cdot$
$a^{2^{10}} \bmod m) \bmod m \cdot$
$a^{2^9} \bmod m) \bmod m \cdot$
$a^{2^5} \bmod m) \bmod m \cdot$
$a^{2^3} \bmod m) \bmod m \cdot$
$a^{2^2} \bmod m) \bmod m \cdot$
$a^{2^0} \bmod m) \bmod m$

The fast exponentiation algorithm computes $a^n \bmod m$ using $O(\log n)$ multiplications $\bmod m$

primality

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

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

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

factorization

If n is composite, it has a factor of size at most $\sqrt{n}$.

euclid's theorem

There are an infinite number of primes.

Proof by contradiction:

Suppose that there are only a finite number of primes:
$p_1, p_2, \ldots, p_n$

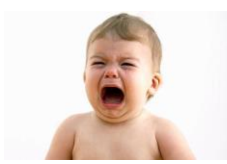
famous algorithmic problems

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

factoring

Factor the following 232 digit number [RSA768]:

1230186684530117755130494958384962720772853
5695953347921973224521517264005072636575187
4520219978646938995647494277406384592519255
7326303453731548268507917026122142913461670
4292143116022212404792747377940806653514195
9745985690214 3413



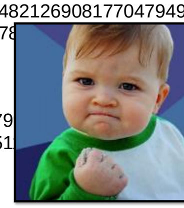
1230186684530117755130494958384962720772853569595334792197322452151726400507263657518745202199786469389956474942774063845925192557326303453731548268507917026122142913461670429214311602221240479274737794080665351419597459856902143413

=

3347807169895689878604416984821269081770479498371376856891243138898288379387 … 17
4308773781446799989

×

3674604366679959042824463379 … 643
430876426760322838157396665 … 968
102700927987363089 17



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) =

gcd and factoring

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

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

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

Factoring is expensive!

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

useful GCD fact

If a and b are positive integers, then
$\gcd(a, b) = \gcd(b, a \bmod b)$

Proof:
By definition $a = (a \text{ div } b) \cdot b + (a \bmod b)$
If $d \mid a$ and $d \mid b$ then $d \mid (a \bmod b)$.
If $d \mid b$ and $d \mid (a \bmod b)$ then $d \mid a$.

euclid's algorithm

Repeatedly use the GCD fact to reduce numbers
until you get $\text{GCD}(x, 0) = x$.

GCD(660,126)

euclid's algorithm

GCD(x, y) = GCD(y, x mod y)

```
int GCD(int a, int b){ /* a >= b, b > 0 */
   int tmp;
   while (b > 0) {
      tmp = a % b;
      a = b;
      b = tmp;
   }
   return a;
}
```

Example: GCD(660, 126)