05 Number Theory and Cryptography

CS201 Discrete Mathematics

Instructor: Shan Chen

Number Theory

- Number theory is a branch of mathematics that explores integers and their properties.
  - It is the basis of many areas, e.g., cryptography, coding theory, computer security, e-commerce, etc.

- At one point, the largest employer of mathematicians in the United States, and probably the world, was the National Security Agency (NSA). ** or No Such Agency?*
  - NSA is the largest spy agency in the US (larger than CIA, Central Intelligence Agency); it is responsible for code design and breaking.

Fun Story

- **Godfrey Harold Hardy** (1877-1947), UK mathematician
  - In his autobiography ***A Mathematician's Apology***, Hardy wrote: "The great modern achievements of applied mathematics have been in relativity and quantum mechanics, and these subjects are, at present, almost as 'useless' as the theory of numbers."
  - If he could see the world now, he would be spinning in his grave :)

Divisibility and Modular Arithmetic

Divisibility

- For integers *a, b* with $a \neq 0$, we say that *a* divides *b* if there is an integer *c* such that $b = ac$, or equivalently *b/a* is an integer.
  - In this case, *a* is a factor or divisor of *b*, and *b* is a multiple of *a*.
- Notation: let $a \mid b$ denote *a* divides *b* (or *b* is divisible by *a*) and let $a \nmid b$ denote *a* does not divide *b* (or *b* is not divisible by *a*).
  - E.g., we have $4 \mid 24$ and $3 \nmid 7$.
- All integers divisible by $d > 0$ can be enumerated as:

  $\ldots, -kd, \ldots, -2d, -d, 0, d, 2d, \ldots, kd, \ldots$
- How many positive integers $\leq n$ are divisible by $d > 0$?
  - Count the number of integers written as kd such that $0 < kd \leq n$. Therefore, there are $\lfloor n/d \rfloor$ such positive integers.

Divisibility Properties

- **Theorem:** Let a, b, c be integers ($a \neq 0$). Then

  (i) if $a \mid b$ and $a \mid c$, then $a \mid (b + c)$

  (ii) if $a \mid b$ then $a \mid bc$ for all integers c

  (iii) if $a \mid b$ and $b \mid c$, then $a \mid c$

  - *proof by definition*

- **Corollary:** If a, b, c are integers ($a \neq 0$) and $a \mid b$, $a \mid c$ hold, then we have $a \mid (mb + nc)$ for any integers m and n.

  - *proved by applying (i) and (ii) of the above Theorem*

The Division Algorithm

- **Theorem:** For any integers a, d with $d > 0$, there exist unique integers q, r, with $0 \leq r < d$, such that $a = d \cdot q + r$.
  - In this case, d is the divisor, a is the dividend, q is the quotient, and r is the remainder.
  - *to be proved in later sections*
- Notation: $q = a \text{ div } d$ and $r = a \bmod d$. * *mod is short for modulo*
- Order of precedence for *mod*: same as multiplication and division
- Example: $17 = 3 \times 5 + 2$
  - $17 \text{ div } 3 = 5$
  - $17 \bmod 3 = 2$

Computing the *mod* Function

- **Theorem:** For integers *a, b, m* with *m* > *0*, we have:
  - *(a + b) mod m = ((a mod m) + (b mod m)) mod m*
  - *ab mod m = (a mod m)(b mod m) mod m*
  - *proof is easy by the following observation*
- Key observation:
  - *a = m (a div m) + (a mod m) = mq + (a mod m)*
- Example:
  - *(78 + 99) mod 5 = ?*

    ((78 mod 5) + (99 mod 5)) mod 5 = (3 + 4) mod 5 = 2
  - *32 x 758 mod 5 = ?*

    (32 mod 5)(758 mod 5) mod 5 = 2 x 3 mod 5 = 1

Arithmetic Modulo *m*

- Let $\boldsymbol{Z}_m = \{0, 1, \ldots, m - 1\}$ be the set of nonnegative integers < m. For $a, b \in \boldsymbol{Z}_m$, addition $+_m$ and multiplication $\cdot_m$ are defined as:
  - $+_m$: $a +_m b = (a + b) \bmod m$
  - $\cdot_m$: $a \cdot_m b = ab \bmod m$
- Examples:
  - $7 +_{11} 9 = ?$

    $(7 + 9) \bmod 11 = 5$
  - $7 \cdot_{11} 9 = ?$

    $7 \cdot 9 \bmod 11 = 8$

Modular Arithmetic Properties

- **Closure:** if $a, b \in \boldsymbol{Z}_m$, then $a +_m b, a \cdot_m b \in \boldsymbol{Z}_m$
- **Associativity:** if $a, b, c \in \boldsymbol{Z}_m$, then

  $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$
- **Identity elements:** if $a \in \boldsymbol{Z}_m$, then $a +_m 0 = a$ and $a \cdot_m 1 = a$
- **Additive inverses:** if $a \neq 0$ and $a \in \boldsymbol{Z}_m$, then $m - a$ is an additive inverse of a modulo m, i.e., $m - a \in \boldsymbol{Z}_m$ and $a +_m (m - a) = 0$
- **Commutativity:** if $a, b \in \boldsymbol{Z}_m$, then $a +_m b = b +_m a$ and $a \cdot_m b = b \cdot_m a$
- **Distributivity:** if $a, b, c \in \boldsymbol{Z}_m$, then

  $a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$

  $(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c)$

Integer Representations and Algorithms

Integer Representations

- There are many ways to represent integers: decimal (base *10*) or binary (base *2*) or octal (base *8*) or hexadecimal (base *16*) or other notations.

- Let $b > 1$ be an integer. Any positive integer n can be expressed uniquely in the form:

$$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0,$$

where k, a_i are nonnegative integers and $0 \leq a_i < b$.

- This representation of n is called the base-b expansion of n, denoted by $(a_k a_{k-1} \cdots a_1 a_0)_b$.

Base-*b* Expansions

- Recall that the base-*b* expansion of $n = (a_k a_{k-1} \cdots a_1 a_0)_b$ means:

  $$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0$$

- Getting the decimal expansion is easy.
  - Examples:

    $(101011111)_2 = 2^8 + 2^6 + 2^4 + 2^3 + 2^2 + 2^1 + 2^0 = 351$

    $(7016)_8 = 7 \cdot 8^3 + 1 \cdot 8 + 6 = 3598$

- Conversions between binary, octal, hexadecimal expansions are also easy.
  - Examples:

    $(101011111)_2 = (101011111)_2 = (537)_8$

    $(7016)_8 = (111000001110)_2 = (111000001110)_2 = (E0E)_{16}$

Constructing Base-*b* Expansions

- Base-*b* expansion can be derived from *mod b* operations:

$$\begin{aligned} n &= a_k b^k + a_{k-1} b^{k-1} + a_{k-2} b^{k-2} + \cdots + a_2 b^2 + a_1 b + a_0 \\ &= b(a_k b^{k-1} + a_{k-1} b^{k-2} + a_{k-2} b^{k-3} + \cdots + a_2 b + a_1) + a_0 \\ &= b(b(a_k b^{k-2} + a_{k-1} b^{k-3} + a_{k-2} b^{k-4} + \cdots + a_2) + a_1) + a_0 \\ &= \cdots \end{aligned}$$

- Algorithm: constructing the base-*b* expansion of an integer *n*

1. Divide *n* by *b* to get $a_0 = n \bmod b$, with $n = bq_0 + a_0$, $0 \le a_0 < b$, then set a_0 as the rightmost digit in the base-*b* expansion of *n*.
2. Divide q_0 by *b* to get $a_1 = q_0 \bmod b$, with $q_0 = bq_1 + a_1$, $0 \le a_1 < b$, then set a_1 as the second digit from the right.
3. Continue this process by successively mod the quotients by *b* until the quotient q_i is 0.

Constructing Base-*b* Expansions

ALGORITHM 1 Constructing Base *b* Expansions.

procedure *base b expansion*(n, b: positive integers with $b > 1$)
$q := n$
$k := 0$
while $q \neq 0$
    $a_k := q$ **mod** b
    $q := q$ **div** b
    $k := k + 1$
return $(a_{k-1}, \ldots, a_1, a_0)$ {$(a_{k-1} \ldots a_1 a_0)_b$ is the base b expansion of n}

- Example: $(12345)_{10} = (30071)_8$

  $12345 = 8 \cdot 1543 + 1$

  $1543 = 8 \cdot 192 + 7$

  $192 = 8 \cdot 24 + 0$

  $24 = 8 \cdot 3 + 0$

  $3 = 8 \cdot 0 + 3$

Binary Addition of Integers

- **Add** $a = (a_{n-1}a_{n-2} \cdots a_1a_0)_2$ and $b = (b_{n-1}b_{n-2} \cdots b_1b_0)_2$
  - start from right to left and maintain a carry bit c
  - $O(n) = O(\max(\log a, \log b))$ bit additions/subtractions
- Note: binary subtraction is similar and also runs in $O(n)$

ALGORITHM 2 Addition of Integers.

procedure *add*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}a_{n-2} \ldots a_1a_0)_2$
and $(b_{n-1}b_{n-2} \ldots b_1b_0)_2$, respectively}
$c := 0$
for $j := 0$ **to** $n - 1$
    $d := \lfloor (a_j + b_j + c)/2 \rfloor$
    $s_j := a_j + b_j + c - 2d$
    $c := d$
$s_n := c$
return $(s_0, s_1, \ldots, s_n)$ {the binary expansion of the sum is $(s_ns_{n-1} \ldots s_0)_2$}

Binary Multiplication of Integers

- **Multiply** $a = (a_{n-1}a_{n-2} \cdots a_1a_0)_2$ by $b = (b_{n-1}b_{n-2} \cdots b_1b_0)_2$
  - $ab = a(b_02^0 + \cdots + b_{n-1}2^{n-1}) = a(b_02^0) + \cdots + a(b_{n-1}2^{n-1})$
  - $O(n^2) = O(\log a \log b)$ bit operations: n rounds of binary shifts and additions

ALGORITHM 3 Multiplication of Integers.

procedure *multiply*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}a_{n-2} \ldots a_1a_0)_2$
and $(b_{n-1}b_{n-2} \ldots b_1b_0)_2$, respectively}
for $j := 0$ **to** $n - 1$
    if $b_j = 1$ **then** $c_j :=$ a shifted j places
    else $c_j := 0$
{$c_0, c_1, \ldots, c_{n-1}$ are the partial products}
$p := 0$
for $j := 0$ **to** $n - 1$
    $p := add(p, c_j)$
return p {p is the value of ab}

Binary Division of Integers

- **Divide** a by $d > 0$: compute $q = a \text{ div } d$ and $r = a \bmod d$
  - $O(q \log a)$ bit operations: q rounds of binary subtractions from a
  - $O(q \log a) = O(2^{\log_2 q} \log d)$ is exponential in the input size $\log_2 ad$ for small $\log_2 d$, e.g., for $\log_2 d < 1/2 \log_2 a$
    ** note that $\log_2 q \approx \log_2 a - \log_2 d \approx \log_2 a + \log_2 d = \log_2 ad$*

ALGORITHM 4 Computing div and mod.

procedure *division algorithm*(a: integer, d: positive integer)
$q := 0$
$r := |a|$
while $r \geq d$
    $r := r - d$
    $q := q + 1$
if $a < 0$ and $r > 0$ **then**
    $r := d - r$
    $q := -(q + 1)$
return (q, r) {$q = a$ **div** d is the quotient, $r = a$ **mod** d is the remainder}

Binary Division of Integers (fast)

- **Divide** a by $d > 0$: compute $q = a \text{ div } d$ and $r = a \bmod d$
  - key observation: $a = 2\lfloor a/2 \rfloor + (a \bmod 2)$ for any $a \geq 0$
  - recursively solve $\lfloor a/2 \rfloor = d \cdot q' + r'$ and use q', r' to compute q, r
  - $O(\log a \cdot \max(\log q, \log d))$ bit operations: $\approx \log_2 a$ iterations of binary shifts/additions/subtractions on q, r (note that $0 \leq r < d$)

procedure *division2* $(a, d \in \mathbb{N},\ d \geq 1)$
if $a < d$
  return $(q, r) = (0, a)$
$(q, r) = division2(\lfloor a/2 \rfloor, d)$
$q = 2q,\ r = 2r$
if a is odd
  $r = r + 1$
if $r \geq d$
  $r = r - d$
  $q = q + 1$
return (q, r)

$\approx \log_2 a$ recursive division2 calls

* There exist more efficient algorithms that run in $O(\log a \log d)$ time

Fast Modular Exponentiation

- Compute $b^n \bmod m$ (where $n = (a_{k-1} \cdots a_1 a_0)_2$)
  - $b^n = b^{a_{k-1} \cdot 2^{k-1} + \cdots + a_1 \cdot 2 + a_0} = b^{a_{k-1} \cdot 2^{k-1}} \cdots b^{a_1 \cdot 2} \cdot b^{a_0}$
  - Compute successively (squared each time): $b \bmod m$, $b^2 \bmod m$, $b^{2^2} \bmod m$, …, $b^{2^{k-1}} \bmod m$, and multiply result by b^{2^i} when $a_i = 1$.
  - $O(\log n (\log m)^2 + \log b \log m)$ bit operations: $b \bmod m$ + $\log_2 n$ rounds of binary multiplications of two $\leq m$ integers then modulo m

ALGORITHM 5 Fast Modular Exponentiation.

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2} \ldots a_1 a_0)_2$,
m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ **to** $k - 1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return $x\{x$ equals b^n **mod** $m\}$

Exercise *(5 mins)*

- Compute $3^{233} \bmod 105$ using fast modular exponentiation.

ALGORITHM 5 Fast Modular Exponentiation.

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2} \ldots a_1a_0)_2$,
m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ **to** $k - 1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return $x\{x$ equals b^n **mod** $m\}$

Exercise *(5 mins)*

- Compute $3^{233} \bmod 105$ using fast modular exponentiation.
- **Solution:**
  - First, construct the binary expansion of 233: $(11101001)_2$
  - Then, apply fast modular exponentiation algorithm to compute the result r as follows:

$$i = 0 : a_0 = 1, 3^{2^0} \bmod 105 = 3, r = 1 \cdot 3 \bmod 105 = 3.$$
$$i = 1 : a_1 = 0, 3^{2^1} \bmod 105 = 3^2 \bmod 105 = 9, r = 3.$$
$$i = 2 : a_2 = 0, 3^{2^2} \bmod 105 = 9^2 \bmod 105 = 81, r = 3.$$
$$i = 3 : a_3 = 1, 3^{2^3} \bmod 105 = 81^2 \bmod 105 = 51, r = 3 \cdot 51 \bmod 105 = 48.$$
$$i = 4 : a_4 = 0, 3^{2^4} \bmod 105 = 51^2 \bmod 105 = 81, r = 48.$$
$$i = 5 : a_5 = 1, 3^{2^5} \bmod 105 = 81^2 \bmod 105 = 51, r = 48 \cdot 51 \bmod 105 = 33.$$
$$i = 6 : a_6 = 1, 3^{2^6} \bmod 105 = 51^2 \bmod 105 = 81, r = 33 \cdot 81 \bmod 105 = 48.$$
$$i = 7 : a_7 = 1, 3^{2^7} \bmod 105 = 81^2 \bmod 105 = 51, r = 48 \cdot 51 \bmod 105 = 33.$$

Primes and Greatest Common Divisors

Primes and Prime Factorization

- **Prime:** a positive integer p that is greater than or equal to 2 and has only two positive factors 1 and p
  - E.g., 13 is a prime, with only two positive factors 1 and 13.
  - We already proved before that there are infinitely many primes.
- **Composite:** a positive integer ≥ 2 that is not a prime
- **Fundamental Theorem of Arithmetic:** Every integer ≥ 2 can be written uniquely as a prime or as the product of multiple primes, where the prime factors are written in nondecreasing order.
  - E.g., $12 = 2 \cdot 2 \cdot 3$
  - The above is also known as the **prime factorization theorem**.
  - It is not hard to see the existence of a prime factorization, but its uniqueness is not straightforward.

Uniqueness of Prime Factorization

- **Lemma:** If p is prime and $p \mid a_1 a_2 \cdots a_n$, then $p \mid a_i$ for some i.
  - *to be proved by induction and Bézout's theorem in later sections*
- **Theorem:** a prime factorization of a positive integer where the primes are in nondecreasing order is unique.
- Proof by contradiction:
  - Suppose that the positive integer n can be written as a product of primes in two distinct ways:

    $n = p_1 p_2 \cdots p_s$ and $n = q_1 q_2 \cdots q_t$
  - Remove all common primes that appear in both factorizations:

    $p_{i_1} p_{i_2} \cdots p_{i_u} = q_{j_1} q_{j_2} \cdots q_{j_v} \neq 1$
  - Since p_{i_1} divides the left side, it must also divide the right side. Then, by **Lemma**, we have $p_{i_1} \mid q_{j_k}$ for some k, which contradicts the assumption that p_{i_1} and q_{j_k} are distinct primes.

Primality Tests

- **Primality test:** an algorithm for determining whether a number is prime or composite
  - Approach 1: test if each integer $2 \le x < n$ divides n.
  - Approach 2: test if each prime number $x < n$ divides n.
  - **Trivial division:** test if each prime number $x \le \sqrt{n}$ divides n. * *why?*

- **Theorem:** If n is composite, then n has a prime divisor $\le \sqrt{n}$.

- Proof: If n is composite, then n has an integer factor a such that $2 \le a < n$. Then, $n = a \cdot b$ and b is an integer such that $2 \le b < n$. If $a > \sqrt{n}$ and $b > \sqrt{n}$, then $a \cdot b > n$, which contradicts $n = a \cdot b$. Therefore, we have $a \le \sqrt{n}$ or $b \le \sqrt{n}$, so n has a divisor $d \le \sqrt{n}$. By the **fundamental theorem of arithmetic**, d is either prime or a product of multiple prime factors. In either case, n has a prime divisor $\le \sqrt{n}$.

The Sieve of Eratosthenes

- How to find all primes ≤ 100?
  - Delete integers divisible by 2
  - Delete integers divisible by 3
  - Delete integers divisible by 5
  - Delete integers divisible by 7

- Why it works?
  - 7 is the largest prime $\leq \sqrt{100}$

TABLE 1 **The Sieve of Eratosthenes.**

Integers divisible by 2 other than 2 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Integers divisible by 3 other than 3 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Integers divisible by 5 other than 5 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Integers divisible by 7 other than 7 receive an underline; integers in color are prime.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Mersenne Primes

- **Mersenne prime:** a prime of the form $2^p - 1$, where p is prime
- Examples:
  - $2^2 - 1 = 3$
  - $2^3 - 1 = 7$
  - $2^5 - 1 = 37$
  - $2^{11} - 1 = 2047 = 23 \cdot 89$ is not a Mersenne prime

Marin Mersenne

- The largest known prime numbers are Mersenne primes.
  - https://www.mersenne.org/

51st Known Mersenne Prime Found!

December 21, 2018 — The Great Internet Mersenne Prime Search (GIMPS) has discovered the largest known prime number, $2^{82,589,933}-1$, having 24,862,048 digits. A computer volunteered by Patrick Laroche from Ocala, Florida made the find on December 7, 2018. The new prime number, also known as M82589933, is calculated by multiplying together 82,589,933 twos and then subtracting one. It is more than one and a half million digits larger than the previous record prime number.

Conjectures on Primes

- **Goldbach's conjecture (*1 + 1*):** Every even integer that is greater than *2* is the sum of two primes.
  - "3 + 4", "3 + 3", "2 + 3" – Y. Wang, 1956
  - "1 + 5" – C. Pan, 1962
  - "1 + 4" – Y. Wang, 1962
  - "1 + 2" – J. Chen, 1973

    Every sufficiently large even integer can be written as the sum of a prime and a semiprime (the product of two primes)

- **Twin-prime conjecture:** There are infinitely many twin primes.
  - A twin prime is a prime number that is either *2* less or *2* more than another prime number, e.g., either of the twin-prime pair *(41, 43)*.

Greatest Common Divisor (GCD)

- Let *a* and *b* be integers, not both *0*. The largest integer *d* such that $d \mid a$ and $d \mid b$ is called the greatest common divisor (GCD) of *a* and *b*, denoted by *gcd(a, b)*.
  - E.g., *gcd(12, -21) = ?*
- One systematic way to find the GCD is prime factorization:

  Let $|a| = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}$ and $|b| = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n}$. Then,

  $$gcd(a, b) = p_1^{min(a_1, b_1)} p_2^{min(a_2, b_2)} \cdots p_n^{min(a_n, b_n)}$$

  - E.g., $12 = 2^2 \cdot 3^1 \cdot 7^0$, $|-21| = 2^0 \cdot 3^1 \cdot 7^1$

    $gcd(12, -21) = 2^0 \cdot 3^1 \cdot 7^0 = 3$
- Two integers *a* and *b* are relatively prime (also called coprime) if their greatest common divisor *gcd(a, b) = 1*.

Least Common Multiple (LCM)

- Let *a* and *b* be non-zero integers. The smallest positive integer that is divisible by both *a* and *b* is called the least common multiple (LCM) of *a* and *b*, denoted by *lcm(a, b)*.
  - E.g., *lcm(12, -21) = ?*
- We can also use factorization to find the LCM systematically:

  Let $|a| = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}$ and $|b| = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n}$. Then,

  $$lcm(a, b) = p_1^{max(a_1, b_1)} p_2^{max(a_2, b_2)} \cdots p_n^{max(a_n, b_n)}$$

  - E.g., $12 = 2^2 \cdot 3^1 \cdot 7^0$, $|-21| = 2^0 \cdot 3^1 \cdot 7^1$

    $lcm(12, -21) = 2^2 \cdot 3^1 \cdot 7^1 = 84$

The Euclidean Algorithm

- Computing the GCD of two integers with prime factorization could be cumbersome and time consuming, since we have to find all prime factors of the two integers.

- Luckily, we have an efficient algorithm, called the Euclidean algorithm. This algorithm has been known since ancient times and is named after the ancient Greek mathematician **Euclid**.

- Example: compute *gcd(287, 91)*

  Step 1: $287 = 91 \cdot 3 + 14$

  Step 2: $91 = 14 \cdot 6 + 7$

  Step 3: $14 = 7 \cdot 2 + 0$

  $gcd(287, 91) = gcd(91, 14) = gcd(14, 7) = 7$

  ** works like magic :)*

The Euclidean Algorithm

ALGORITHM 1 The Euclidean Algorithm.

procedure *gcd*(*a*, *b*: positive integers)
$x := a$
$y := b$
while $y \neq 0$
    $r := x$ **mod** y
    $x := y$
    $y := r$
return $x\{\gcd(a, b) \text{ is } x\}$

number of mod operations: O(log min(a, b))

** proof is left as an assignment problem*

- Example: compute *gcd(287, 91)*

  Step 1: *287* = *91* · 3 + *14*

  Step 2: *91* = *14* · 6 + *7*

  Step 3: *14* = *7* · 2 + *0*

  gcd(287, 91) = gcd(91, 14) = gcd(14, 7) = 7

Correctness of Euclidean Algorithm

- **Lemma:** Let $a = b \cdot q + r$, where a, b, q and r are integers, then $gcd(a, b) = gcd(b, r)$.

- Proof:

  - For any d such that $d \mid a$ and $d \mid b$, d also divides $a - b \cdot q = r$. Hence, any common divisor of a and b must also be a common divisor of b and r. This implies $gcd(a, b) \leq gcd(b, r)$.

  - For any d such that $d \mid b$ and $d \mid r$, d also divides $b \cdot q + r = a$. Hence, any common divisor of b and r must also be a common divisor of a and b. This implies $gcd(a, b) \geq gcd(b, r)$.

  - Therefore, $gcd(a, b) = gcd(b, r)$.

Correctness of Euclidean Algorithm

- Proof (correctness of the Euclidean algorithm):
  - Suppose that a and b are positive integers with $a \geq b$. Let $r_0 = a$ and $r_1 = b$. We obtain the following divisions from the algorithm:

$$
\begin{aligned}
r_0 &= r_1 q_1 + r_2 && 0 \leq r_2 < r_1, \\
r_1 &= r_2 q_2 + r_3 && 0 \leq r_3 < r_2, \\
&\vdots \\
r_{n-2} &= r_{n-1} q_{n-1} + r_n && 0 \leq r_n < r_{n-1}, \\
r_{n-1} &= r_n q_n.
\end{aligned}
$$

- The number of divisions is finite because $r_0 > r_1 > \cdots > r_{n-1} > r_n \geq 0$.
- Reall **Lemma**: "Let $a = b \cdot q + r$, where a, b, q and r are integers, then $gcd(a, b) = gcd(b, r)$." We have:

  $gcd(a, b) = gcd(r_0, r_1) = \cdots = gcd(r_{n-1}, r_n) = gcd(r_n, 0) = r_n$

Bézout's Theorem

- **Bézout's theorem:** If a and b are positive integers, then there exist integers s and t such that $gcd(a, b) = s \cdot a + t \cdot b$.
  - s, t are called Bézout coefficients of a and b
  - $gcd(a, b) = s \cdot a + t \cdot b$ is called Bézout's identity
- **Solving** s, t: (two-pass) Euclidean algorithm
  - E.g., $gcd(503, 286) = 1 = 29 \cdot 503 - 51 \cdot 286$

$$503 = 1 \cdot 286 + 217$$
$$286 = 1 \cdot 217 + 69$$
$$217 = 3 \cdot 69 + 10$$
$$69 = 6 \cdot 10 + 9$$
$$10 = 1 \cdot 9 + 1$$

$$\begin{aligned} 1 &= 10 - 1 \cdot 9 \\ &= 7 \cdot 10 - 1 \cdot 69 \\ &= 7 \cdot 217 - 22 \cdot 69 \\ &= 29 \cdot 217 - 22 \cdot 286 \\ &= 29 \cdot 503 - 51 \cdot 286 \end{aligned}$$

in reverse order: substitute for the smaller number at each step

- More efficient way: (one-pass) extended Euclidean algorithm
  - *see the textbook for details (to be proved in later sections)*

Exercise *(5 mins)*

- Find Bézout coefficients of *561* and *234*.

- **Solving** *s, t*: (two-pass) Euclidean algorithm
  - E.g., *gcd(503, 286)* = *1* = *29* · *503* – *51* · *286*

$$
\begin{aligned}
503 &= 1 \cdot 286 + 217 & 1 &= 10 - 1 \cdot 9 \\
286 &= 1 \cdot 217 + 69 & &= 7 \cdot 10 - 1 \cdot 69 \\
217 &= 3 \cdot 69 + 10 & &= 7 \cdot 217 - 22 \cdot 69 \\
69 &= 6 \cdot 10 + 9 & &= 29 \cdot 217 - 22 \cdot 286 \\
10 &= 1 \cdot 9 + 1 & &= 29 \cdot 503 - 51 \cdot 286
\end{aligned}
$$

in reverse order: substitute for the smaller number at each step

Exercise *(5 mins)*

- Find Bézout coefficients of *561* and *234*.
- **Solution:**
  - Apply Euclidean algorithm to find *gcd(561, 234)* = *3.*
  - Then, reverse the equations to find Bézout coefficients: *-5* and *12*

$$
\begin{aligned}
561 &= 2 \cdot 234 + 93 \\
234 &= 2 \cdot 93 + 48 \\
93 &= 1 \cdot 48 + 45 \\
48 &= 1 \cdot 45 + 3.
\end{aligned}
$$

$$
\begin{aligned}
3 &= 1 \cdot 48 - 1 \cdot 45 \\
&= 1 \cdot 48 - 1 \cdot (93 - 48) \\
&= 2 \cdot 48 - 1 \cdot 93 \\
&= 2 \cdot (234 - 2 \cdot 93) - 1 \cdot 93 \\
&= 2 \cdot 234 - 5 \cdot 93 \\
&= 2 \cdot 234 - 5 \cdot (561 - 2 \cdot 234) \\
&= 12 \cdot 234 - 5 \cdot 561.
\end{aligned}
$$

Corollaries of Bézout's Theorem

- **Corollary 1:** If a, b, c are positive integers such that $gcd(a, b) = 1$ and $a \mid bc$, then $a \mid c$.

- Proof: By **Bézout's theorem**, there exist integers s and t such that $gcd(a, b) = 1 = s \cdot a + t \cdot b$. Multiply c on both sides:

  $$c = s \cdot a \cdot c + t \cdot b \cdot c$$

  Since $a \mid bc$, we have $a \mid tbc$. It is also clear that $a \mid sac$.

  Therefore, we have $a \mid (sac + tbc) = c$.

- **Corollary 2:** If p is prime and $p \mid a_1 a_2 \cdots a_n$, then $p \mid a_i$ for some i.
  - this was used to prove the uniqueness of prime factorization
  - *to be proved by induction in later sections*

Congruences

Congruences

- **Definition:** If *a* and *b* are integers and *m* is a positive integer, then *a* is congruent to *b* modulo *m* if and only if $m \mid (a - b)$, denoted by $a \equiv b \pmod{m}$. ** in Chinese this is called "同余"*
  - Alternatively, we say *a* and *b* are congruent modulo *m*.
  - $a \equiv b \pmod{m}$ is called a congruence and *m* is its modulus.

- Examples:
  - $15 \equiv 3 \pmod{6}$
  - $-1 \equiv 11 \pmod{6}$

- **Corollary:** The integers *a* and *b* are congruent modulo *m* if and only if there is an integer *k* such that $a = b + km$.
  - *prove the "if" part and "only if" part*

Notation: *mod m* vs ≡ *(mod m)*

- $a = b \bmod m$ and $a \equiv b \pmod m$ are different:
  - $\bmod m$ denotes a function "*mod m*": $\boldsymbol{Z} \rightarrow \boldsymbol{Z}_m = \{0, 1, \ldots, m-1\}$.
  - $\equiv \pmod m$ is a relation between two integers.

- **Theorem:** Let *a* and *b* be integers, and *m* be a positive integer. Then, $a \equiv b \pmod m$ if and only if $a \bmod m = b \bmod m$.
  - *prove the "if" part and "only if" part*

Congruences of Sums and Products

- **Theorem:** Let *m* be a positive integer. If $a \equiv b \pmod m$ and $c \equiv d \pmod m$, then $a + c \equiv b + d \pmod m$ and $ac \equiv bd \pmod m$
  - *proof by definition*
- Examples: for $a \equiv b \bmod m$ and $c > 0$, are the following true?
  - $ac \equiv bc \pmod m$?

    True
  - $a + c \equiv b + c \pmod m$?

    True
  - $a - c \equiv b - c \pmod m$?

    True
  - $a / c \equiv b / c \pmod m$? * assume $c \mid a$ and $c \mid b$

    False, e.g., $14 \equiv 8 \pmod 6$ but $14 / 2 = 7 \not\equiv 4 = 8 / 2 \pmod 6$

Dividing Congruences by an Integer

- **Theorem:** Let *m* be a positive integer and let *a, b, c* be integers. If $ac \equiv bc \pmod{m}$ and $\gcd(c, m) = 1$, then $a \equiv b \pmod{m}$.

- Proof: By definition of $ac \equiv bc \pmod{m}$, we have $m \mid (ac - bc)$, i.e., $m \mid c(a - b)$. Since $\gcd(c, m) = 1$, it follows from **Corollary 1 of Bézout's theorem** that $m \mid (a - b)$.

- Example: consider $20 \equiv 56 \pmod{9}$ and $\gcd(4, 9) = 1$
  - Dividing the congruence by *4* on both sides, we have:
    $$20 / 4 = 5 \equiv 14 = 56 / 4 \pmod{9}$$

Modular Multiplicative Inverse

- An integer $\bar{a}$ such that $\bar{a}a \equiv 1 \pmod{m}$ is said to be a modular multiplicative inverse (or inverse for simplicity) of a modulo m.
- **Q:** When does an inverse of a modulo m exist?
- **Theorem:** If $\gcd(a, m) = 1$ and $m > 1$, then there exists an inverse of a modulo m. Furthermore, the inverse is unique modulo m.
- Proof: Since $\gcd(a, m) = 1$, by **Bézout's theorem** there exist integers s and t such that $s \cdot a + t \cdot m = 1$, i.e., $sa + tm \equiv 1 \pmod{m}$. Since we have $tm \equiv 0 \pmod{m}$, it follows that $sa \equiv 1 \pmod{m}$. This means that s is an inverse of a modulo m.
  - *proof of uniqueness is left as an assignment problem*
- Actually, for $m > 1$, an inverse of a exists if and only if $\gcd(a, m) = 1$. That is, if $\gcd(a, m) > 1$, then there exists no inverse of a modulo m.
  - *proof is left as an assignment problem*

How to Find an Inverse?

- Approach: find Bézout coefficients by applying the (extended) Euclidean algorithm.
  - *gcd(a, m) = 1 = sa + tm* implies that *s* is the inverse of *a* modulo *m*.
- Example: find a multiplicative inverse of *286* modulo *503*

$$
\begin{aligned}
503 &= 1 \cdot 286 + 217 & 1 &= 10 - 1 \cdot 9 \\
286 &= 1 \cdot 217 + 69 & &= 7 \cdot 10 - 1 \cdot 69 \\
217 &= 3 \cdot 69 + 10 & &= 7 \cdot 217 - 22 \cdot 69 \\
69 &= 6 \cdot 10 + 9 & &= 29 \cdot 217 - 22 \cdot 286 \\
10 &= 1 \cdot 9 + 1 & &= 29 \cdot 503 - 51 \cdot 286
\end{aligned}
$$

** see the textbook for the use of the extended Euclidean algorithm*

Linear Congruences

- **Linear congruence:** a congruence of the form $ax \equiv b \pmod{m}$, where m is a positive integer, a, b are integers and x is a variable.
- The solutions to a linear congruence $ax \equiv b \pmod{m}$ are all integers x that satisfy the congruence.
- Linear congruences have been studied since ancient times.

- About 1500 years ago, the Chinese mathematician **Sun-Tsu** asked: "There are certain things whose number is unknown. If we count them by threes, we have two left over; by fives, we have three left over; and by sevens, two are left over. How many things are there?"

  有物不知其数，三三数之剩二，五五数之剩三，七七数之剩二。问物几何？

- Translation: $x \equiv 2 \pmod{3}$, $x \equiv 3 \pmod{5}$, $x \equiv 2 \pmod{7}$, $x = ?$

Solving Linear Congruences

- **Q:** How to solve a linear congruence $ax \equiv b \pmod m$?

- **Solution:** If an inverse of a modulo m exists, say $\bar{a}$, then one can solve $ax \equiv b \pmod m$ for x by multiplying $\bar{a}$ on both sides. That is, $x \equiv \bar{a}ax \equiv \bar{a}b \pmod m$.
  - Note that $x \equiv \bar{a}ax \pmod m$ follows from $1 \equiv \bar{a}a \pmod m$.

- Example:
  - What are the solutions of the congruence $3x \equiv 4 \pmod 7$?
  - **Solution:** First, find an inverse of 3 modulo 7, e.g., -2 is an inverse because $3 \cdot -2 = -6 \equiv 1 \pmod 7$. Then, multiply both sides of the congruence by -2, we have $x \equiv -6x \equiv -8 \equiv 6 \pmod 7$.

- **Q:** What if the inverse of a modulo m does not exist, i.e., when $gcd(a, m) = d > 1$?

Number of Congruence Solutions

- **Theorem:** Let $d = gcd(a, m)$. The linear congruence $ax \equiv b \pmod m$ has solutions if and only if $d \mid b$. If $d \mid b$, then there are exactly d solutions in $\boldsymbol{Z}_m$.
  - Let $m' = m / d$. If x_0 is a solution, then the other $d - 1$ solutions are $x_0 +_m m'$, $x_0 +_m 2m'$, …, $x_0 +_m (d - 1)m'$. ** recall that $+_m$ denotes modular addition*
- Proof: ("only if" + "if" + "exactly d solutions (# = d)")
  - "only if": If x_0 is one of the solutions, then $ax_0 \equiv b \pmod m$, i.e., $ax_0 - b = km$. Therefore, $ax_0 - km = b$. Since $d \mid a$ and $d \mid m$, we have $d \mid ax_0 - km$, i.e., $d \mid b$.
  - "if": By **Bézout's theorem**, there are integers s, t such that $d = s \cdot a + t \cdot m$. Suppose $d \mid b$, then $b = kd$ for some integer k. Therefore, $b = kd = ksa + ktm$. Then, we have $x_0 = ks$ is a solution because we have $ax_0 = aks \equiv b \pmod m$.
  - "# = d": For any $ax_i \equiv b \pmod m$ and $ax_j \equiv b \pmod m$ we have $m \mid a(x_j - x_i)$. Dividing both sides by d yields $(m / d) \mid (x_j - x_i)(a / d)$. Recall $d = gcd(a, m)$, so $gcd(m / d, a / d) = 1$. By **Corollary 1 of Bézout's theorem**, $(m / d) \mid (x_j - x_i)$. This means that there are at most d distinct solutions in $\boldsymbol{Z}_m$. It is not hard to check that if x_0 is a solution then $x_0 +_m km'$ $(k = 1, …, d - 1)$ is also a solution.

Example

- Does the congruence $3x \equiv 4 \pmod 6$ have solutions?
  - ***No***, because $\gcd(3, 6) = 3$ does not divide 4.
- How to find the solutions to $6x \equiv 8 \pmod{22}$?
  - First, apply Euclidean algorithm to compute $d = \gcd(6, 22) = 2$.
  - The congruence $6x \equiv 8 \pmod{22}$ has solutions since $d \mid b$, i.e., $2 \mid 8$.
  - Then, apply Euclidean algorithm to find Bézout's identity:
    $$d = \gcd(6, 22) = 2 = 4 \cdot 6 - 1 \cdot 22$$
  - Multiply $b / d = 8 / 2 = 4$ on both sides yielding: $8 = 16 \cdot 6 - 4 \cdot 22$. Therefore, $6 \cdot 16 \equiv 8 \pmod{22}$ and $x_0 = 16$ is a solution.
  - There are totally $d = 2$ solutions in $\boldsymbol{Z}_{22}$. Let $m' = m / d = 22 / 2 = 11$. The other solution is $x_1 = (16 + 11) \bmod 22 = 5$.

The Chinese Remainder Theorem

- **The Chinese Remainder Theorem:** 中国剩余定理 Let $m_1, m_2, \cdots, m_n$ be pairwise coprime positive integers ≥ 2 and let $a_1, a_2, \cdots, a_n$ be arbitrary integers. Then, the system of congruences

  $x \equiv a_1 \pmod{m_1}$

  $x \equiv a_2 \pmod{m_2}$

  ...

  $x \equiv a_n \pmod{m_n}$

  has a unique solution modulo $M = m_1 m_2 \cdots m_n$.

- Proof of solution existence (constructive proof):
  Let $M_k = M / m_k$ for $k = 1, \ldots, n$. Since $\gcd(M_k, m_k) = 1$, there exists an integer y_k (as an inverse of M_k modulo m_k) such that $M_k y_k \equiv 1 \pmod{m_k}$.
  Construct $x = a_1 M_1 y_1 + a_2 M_2 y_2 + \cdots + a_n M_n y_n$. It is not hard to check that x is a solution to the above system of n congruences.

  ** proof of uniqueness is left as an assignment problem*

Example

- Example:

$$x \equiv 2 \pmod 3$$
$$x \equiv 3 \pmod 5$$
$$x \equiv 2 \pmod 7$$

三人同行七十稀，五树梅花廿一枝，
七子团圆正月半，除百零五便得知。
-- 程大位 《算法统要》 (1593年)

- **Solution** (Chinese remainder theorem):
  - Let $M = 3 \cdot 5 \cdot 7 = 105$
  - $M_1 = M / 3 = 35$, $M_2 = M / 5 = 21$, $M_3 = M / 7 = 15$

$35 \cdot 2 \equiv 1 \pmod 3$ $\quad y_1 = 2$

$21 \cdot 1 \equiv 1 \pmod 5$ $\quad y_2 = 1$

$15 \cdot 1 \equiv 1 \pmod 7$ $\quad y_3 = 1$

  - $x = 2 \cdot 35 \cdot 2 + 3 \cdot 21 \cdot 1 + 2 \cdot 15 \cdot 1 \equiv 233 \equiv 23 \pmod{105}$

** What if the moduli are not pairwise coprime? (see assignment)*

Back Substitution

- Example:

$$x \equiv 2 \pmod 3$$
$$x \equiv 3 \pmod 5$$
$$x \equiv 2 \pmod 7$$

- **Solution** (back substitution):
  - There exists an integer t such that $x = 3t + 2$.
  - Substitute this into the 2nd congruence: $3t + 2 \equiv 3 \pmod 5$ and solve it as $t \equiv 2 \pmod 5$, *i.e.*, $t = 5u + 2$ for some integer u.
  - Substitute this back into $x = 3t + 2$ shows $x = 15u + 8$.
  - Substitute this into the 3rd congruence: $15u + 8 \equiv 2 \pmod 7$ and we can solve it as $u \equiv 1 \pmod 7$, *i.e.*, $u = 7v + 1$ for some integer v.
  - Substitute this back into $x = 15u + 8$ tells us $x = 105v + 23$.
  - Therefore, $x \equiv 23 \pmod{105}$.

Fermat's Little Theorem

- **Fermat's Little Theorem:** If p is prime and $a \not\equiv 0 \pmod{p}$, then

$$a^{p-1} \equiv 1 \pmod{p}$$

Furthermore, for every integer a we have

$$a^{p} \equiv a \pmod{p}$$

  - *the proof is left as an assignment problem*

- Example: $7^{222} \equiv ? \pmod{11}$
  - $7^{222} = 7^{10 \cdot 22 + 2} = (7^{10})^{22} 7^2 \equiv 1^{22} \cdot 49 \equiv 5 \pmod{11}$

- **Q:** What is **Fermat's Last Theorem**? 费马大定理
  - *The equation $a^n + b^n = c^n$, where the integer $n > 2$, has no integer solutions a, b, c such that $abc \neq 0$.* * *the first proof found in 1990s*

Euler's Theorem

- Euler's totient function $\phi(n)$ maps a positive integer n to the number of positive integers coprime to n in $\boldsymbol{Z}_n$.

- Examples: (p, q are prime)
  - $\phi(p) = p - 1$
  - $\phi(pq) = (p - 1)(q - 1)$
  - $\phi(p^i) = p^i - p^{i-1}$

  ** count $\phi(n)$ by excluding the integers divisible by n's prime factors*

- **Euler's theorem:** Let a, n be positive coprime integers. Then

  $$a^{\phi(n)} \equiv 1 \pmod{n}$$

  - when n is prime, this becomes **Fermat's little theorem** for $a > 0$
  - *the proof is very similar to that of Fermat's little theorem*

Primitive Roots Modulo a Prime

- A primitive root modulo a prime p is an integer $r \in \boldsymbol{Z}_p$ such that every nonzero element in $\boldsymbol{Z}_p$ is a power of r modulo p.

- Examples:
  - Is *3* is a primitive root modulo *5*?

    Yes, $3 \equiv 3^1 \pmod 5$, $4 \equiv 3^2 \pmod 5$, $2 \equiv 3^3 \pmod 5$, $1 \equiv 3^4 \pmod 5$

  - Is *2* a primitive root modulo *7*?

    No, $2 \equiv 2^1 \pmod 7$, $4 \equiv 2^2 \pmod 7$, $1 \equiv 2^3 \pmod 7$, $2 \equiv 2^4 \pmod 7$…

    ** already cycles so will never reach other numbers: 3, 5, 6*

Primitive Roots in General

- A primitive root modulo a prime p is an integer $r \in \boldsymbol{Z}_p$ such that every nonzero element in $\boldsymbol{Z}_p$ is a power of r modulo p.

- In general, consider an arbitrary integer $n \geq 2$ and

$$\boldsymbol{Z}_n^* = \{k \in \boldsymbol{Z}_n \mid gcd(k, n) = 1\}$$

  A primitive root modulo n is an integer $r \in \boldsymbol{Z}_n^*$ such that every element in $\boldsymbol{Z}_n^*$ is a power of r modulo n.

  - Note that $\boldsymbol{Z}_p^*$ (for prime p) consists of all nonzero elements of $\boldsymbol{Z}_p$.

- **Theorem:** There exists a primitive root modulo n $(n \geq 2)$ if and only if $n = 2, 4, p^k$ or $2p^k$, where p is an odd prime and $k \in \boldsymbol{Z}^+$.

  - *the proof is advanced and beyond the scope of this course*

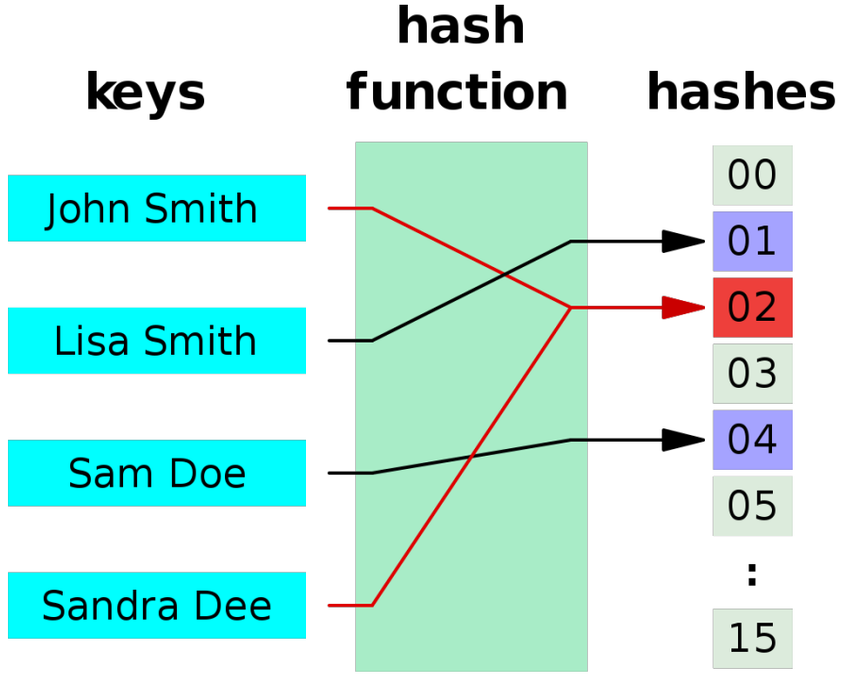
- Primitive roots are very useful in cryptography (as shown later).

Applications of Modular Arithmetic

Hash Functions

- A hash function *h* is a function that maps data of arbitrary length to fixed-length values. The input data is sometimes called keys and the values returned by a hash function are often called hash values, hash codes, digests or simply hashes.
  - Example use case: hashing the identifiers (keys) of data records to their memory locations (hash values)
  - Example hash function: $h(k) = k \bmod m$
- How to handle hash collisions? (two keys mapped to the same hash)
  - move to the next available hash value
  - add a secondary structure

    e.g., hashes pointing to a linked list



SUSTech

Pseudorandom Number Generators

- Pseudorandom numbers are generated by systematic methods to approximate truly random numbers. A pseudorandom number generator (PRNG) is an algorithm for generating them.
  - Motivation: true random numbers are hard to get.
  - PRNGs are widely used in simulation and cryptography.
- One of the widely used PRNGs is the linear congruential method:
  - choose *4* numbers: modulus m, multiplier a, increment c, seed x_0
  - generate a sequence of pseudorandom numbers $\{x_n\}$ in $\boldsymbol{Z}_m$:

$$x_{n+1} = (ax_n + c) \bmod m$$

  - Example: $m = 2^{31} - 1$, $a = 7^5$, $c = 0$ * generates all $m - 1$ numbers
  - Note: This method is not used in sensitive tasks (e.g., for large simulations or cryptography), because it fails to satisfy some important statistical properties of truly random numbers.

Check Bits/Digits

- Congruences can be used to check for errors in bit/digit strings.

- A parity check bit is an extra bit appended to a data block that is stored or transmitted. The parity check bit x_{n+1} for the bit string $x_1x_2 \cdots x_n$ is defined as:

$$x_{n+1} = (x_1 + x_2 + \cdots + x_n) \bmod 2$$

- All books are identified by an International Standard Book Number (ISBN-10), a 10-digit code $x_1x_2 \cdots x_{10}$, assigned by the publisher. An ISBN-10 is valid if and only if the check digit x_{10} satisfies:

$$x_{10} = (x_1 + 2x_2 + \cdots + 9x_9) \bmod 11$$

  - What is the check digit for the ISBN-10 starting with *007288008*?

    2

  - Is *084930149X* a valid ISBN-10 (where $X = 10$)?

    No

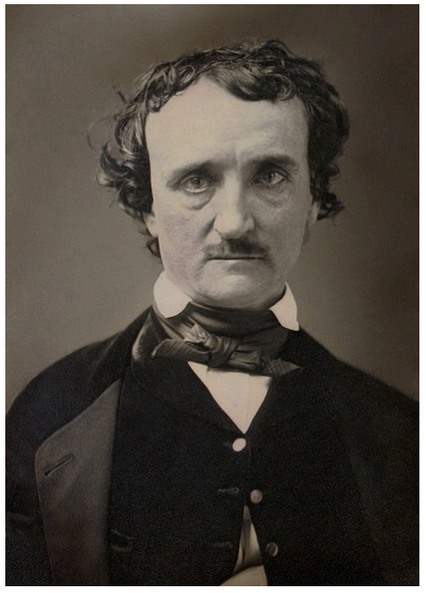
Classical Cryptography

Cryptography and Number Theory

- Roughly, cryptography is the subject of transforming information so that it cannot be easily recovered without special knowledge.
  - "Cryptography is the practice and study of techniques for secure communication in the presence of third parties called adversaries." – **Ronald L. Rivest** (Turing Award winner)
- Number theory plays an important role in cryptography:
  - classical ciphers (before modern cryptography)
  - public-key cryptographic systems

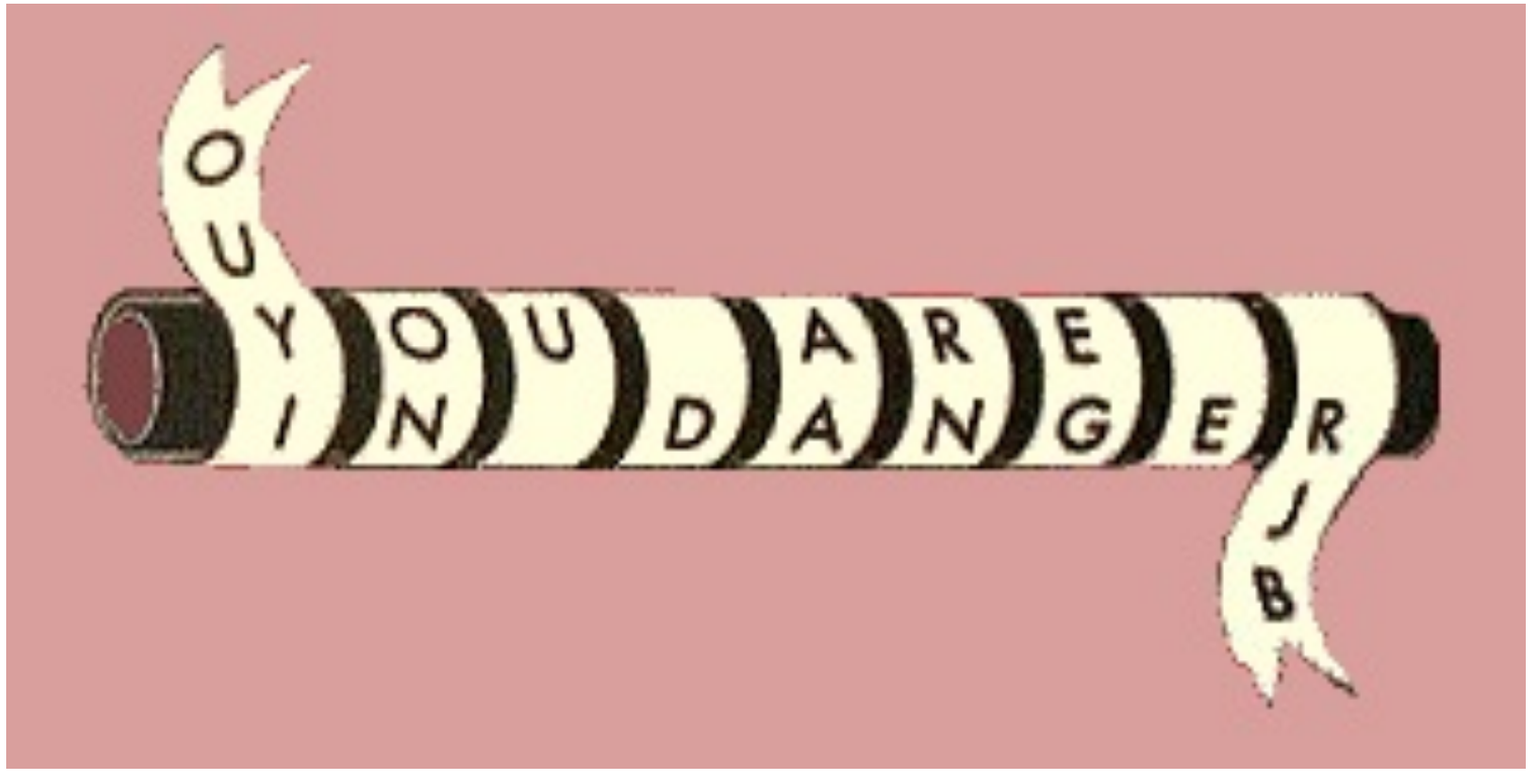
Origin of Cryptography

- History of almost 4000 years (from 1900 B.C.)
- Cryptography = kryptos (secret) + graphos (writing)
- This term was first used in the short story ***The Gold-Bug***, by **Edgar Allan Poe** (1809~1849).
- “Human ingenuity cannot concoct a cipher which human ingenuity cannot resolve.” – 1841
  - *this is believed false in modern cryptography*



Classical Cryptography

- In 405 BC, the Spartan general **Lysander** was sent a coded message written on the inside of a servant's belt.



SUSTech

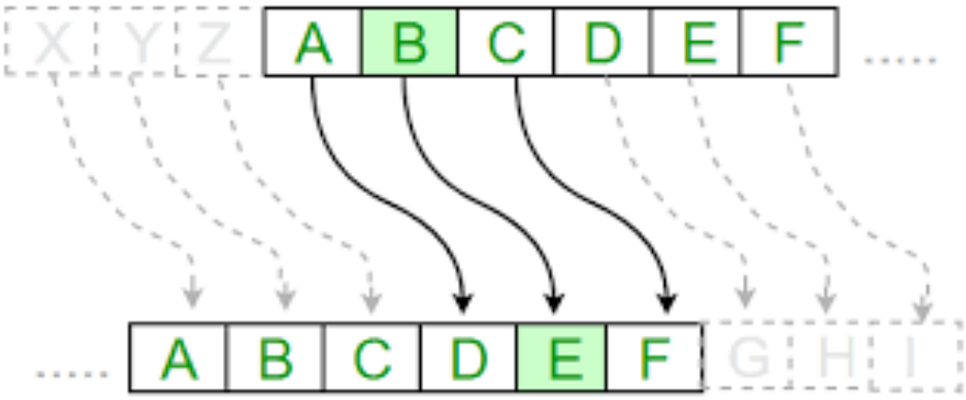
Classical Cryptography

- The Greeks invented a cipher which changed letters to numbers. A form of this code was still being used during World War I.

	1	2	3	4	5
1	A	B	C	D	E
2	F	G	H	I/J	K
3	L	M	N	O	P
4	Q	R	S	T	U
5	V	W	X	Y	Z

Classical Cryptography

- **Caesar cipher** (named after Roman general **Julius Caesar**)



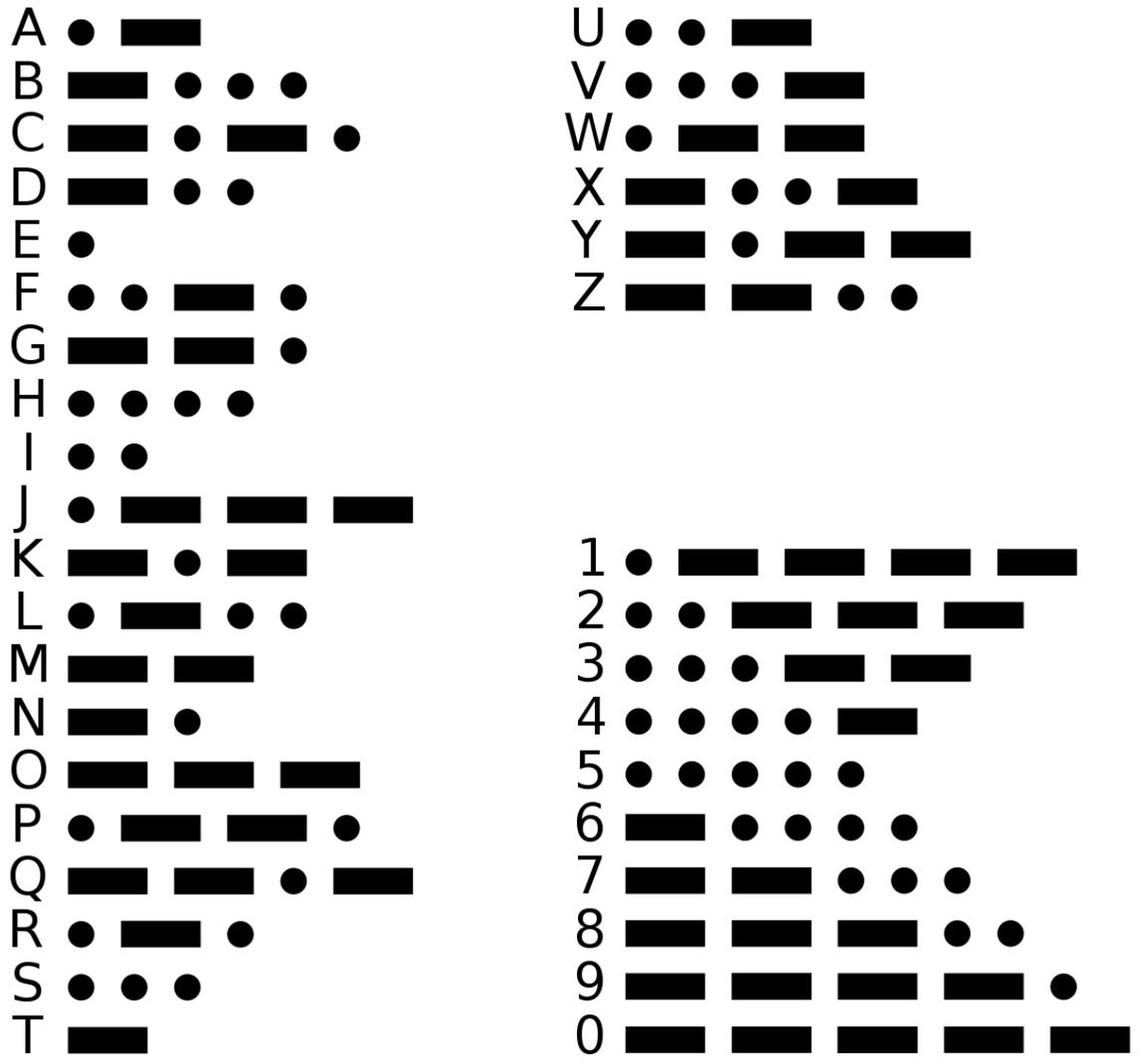
VENI, VIDI, VICI

YHQL YLGL YLFL

SUSTech

Classical Cryptography

- **Morse code:** created by **Samuel Morse** in 1838



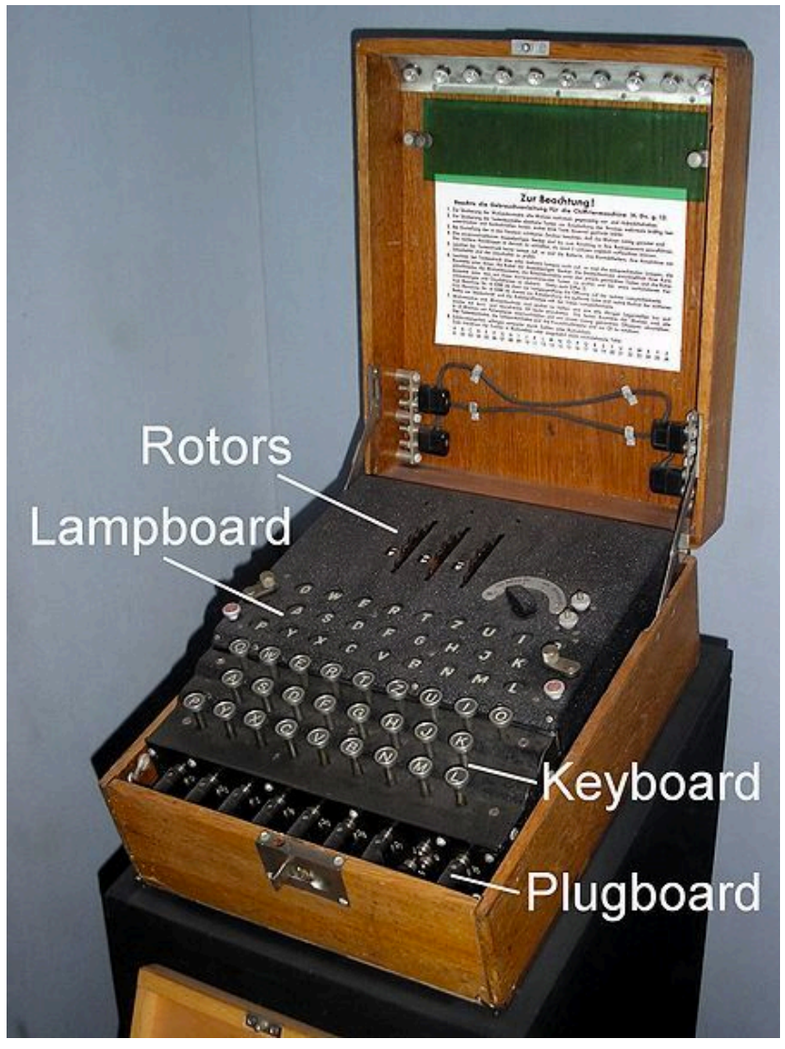
SUSTech

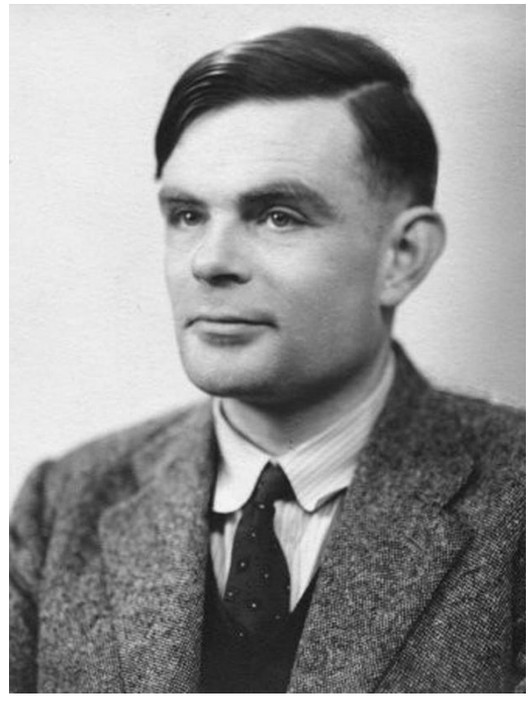
Classical Cryptography

- Cryptograms from the Chinese gold bars
  - http://www.iacr.org/misc/china/china.html

Classical Cryptography

- Enigma: German coding machine in World War II.





Alan Turing
(1912-1954)

Movie: ***The Imitation Game***

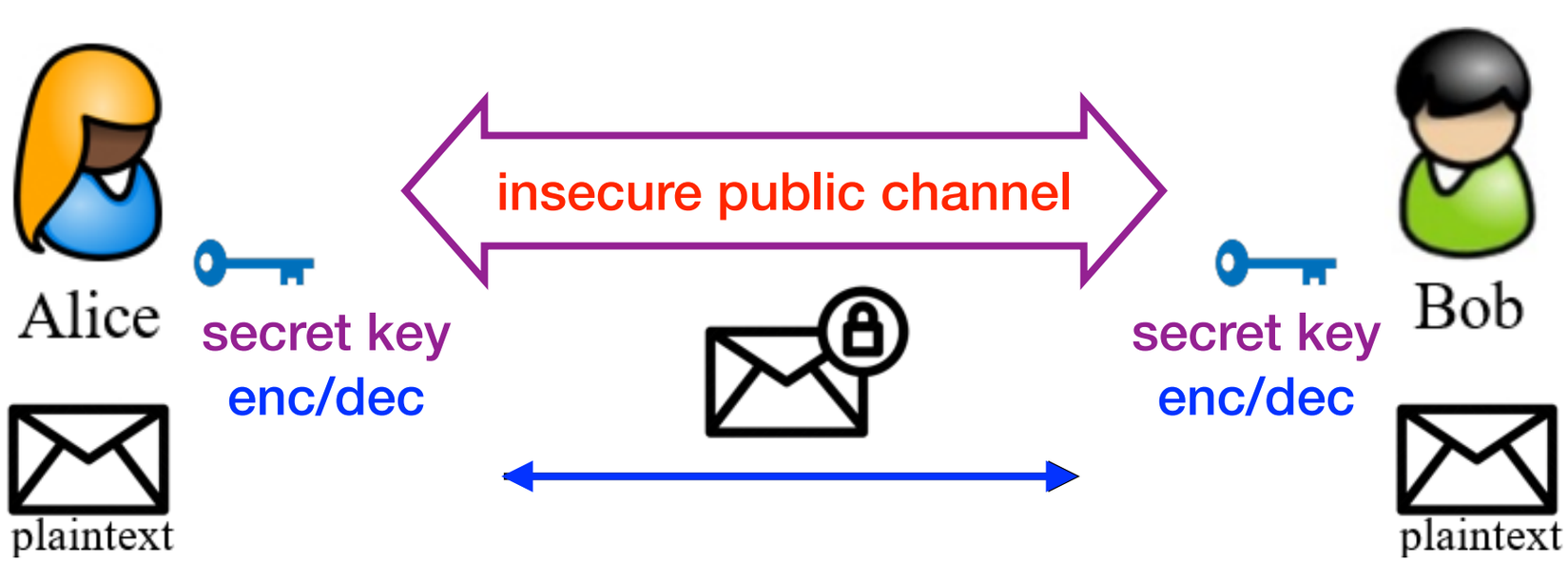
Modern Cryptography

- **Modern cryptography** (since 1970s) makes extensive use of math and mainly consists of two parts:
  - Symmetric cryptography (also called secret-key cryptography)
  - Asymmetric cryptography (also called public-key cryptography)
- **Kerckhoffs's principle** (stated by **Auguste Kerckhoffs** in 1883): a cryptographic system (or simply cryptosystem) should be secure, even if everything about the system, except the key, is public knowledge.
  - security through obscurity does not work
- Next, we will briefly introduce modern cryptography.

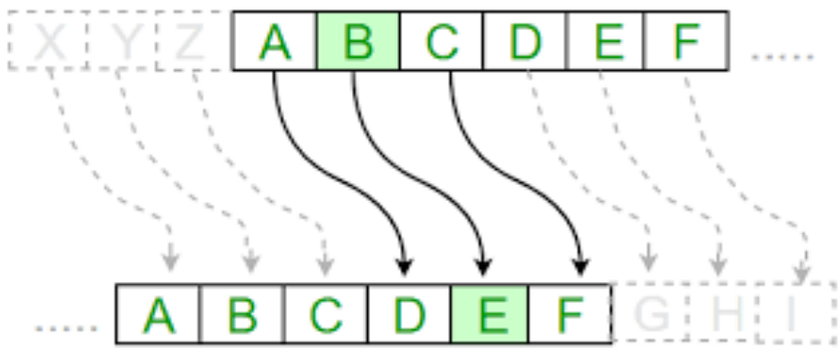
Symmetric Cryptography

Symmetric Cryptography

- **Symmetric cryptography** focuses on cryptosystems where the message sender and receiver share the same secret key for both encryption and decryption.
  - also known as **secret-key cryptography**



Shift Cipher



- Key: *k = 0,1, …, 25*
- Encryption: encrypt *m* as *(m + k) mod 26*
- Decryption: decrypt *c* as *(c – k) mod 26*
- Example: *k = 2*
  - plaintext: SEND RE I NFORCEMENT
  - ciphertext: UGPF TGKPHQTEGOGPV
- **Drawback:** only *26* possible keys!

Substitution Cipher

- Key: table mapping each letter to another letter

A	B	C																					Z
V	R	E																					D

- Encryption & Decryption: letter by letter according to table

- Number of possible keys: $26! \approx 4 \times 10^{26}$

- However, substitution cipher is still insecure!

- **Drawback:** can recover plaintext by analyzing letter frequencies

Substitution Cipher: Attack

Table 1: Relative frequencies of the letters of the English language

Letter	Relative Frequency (%)	Letter	Relative Frequency (%)
a	8.167	n	6.749
b	1.492	o	7.507
c	2.782	p	1.929
d	4.253	q	0.095
e	12.702	r	5.987
f	2.228	s	6.327
g	2.015	t	9.056
h	6.094	u	2.758
i	6.966	v	0.978
j	0.153	w	2.360
k	0.772	x	0.150
l	4.025	y	1.974
m	2.406	z	0.074

Substitution Cipher: Attack

Table 2: Number of Diagraphs Expected in 2,000 Letters of English Text

th	-	50	at	-	25	st	-	20
er	-	40	en	-	25	io	-	18
on	-	39	es	-	25	le	-	18
an	-	38	of	-	25	is	-	17
re	-	36	or	-	25	ou	-	17
he	-	33	nt	-	24	ar	-	16
in	-	31	ea	-	22	as	-	16
ed	-	30	ti	-	22	de	-	16
ne	-	30	to	-	22	rt	-	16
ha	-	26	it	-	20	ve	-	16

Table 3: The 15 Most Common Trigraphs in the English Language

1	-	the	6	-	tio	11	-	edt
2	-	and	7	-	for	12	-	tis
3	-	tha	8	-	nde	13	-	oft
4	-	ent	9	-	has	14	-	sth
5	-	ion	10	-	nce	15	-	men

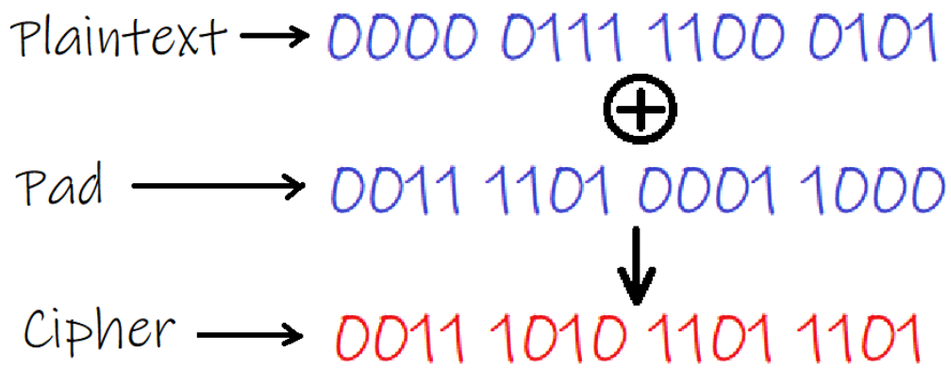
Substitution Cipher: Attack

- Ciphertext: LIVITCSWPIYVEWHEVSRIQMXLEYVEOIEWHRXEXIP FEMVEWHKVSTYLXZIXLIKIIXPIJVSZEYPERRGERIMWQLMGLM XQERIWGPSRIHMXQEREKI

- Frequency analysis:
  - I – most common letter I = e
  - LI – most common pair L = h
  - XLI – most common triple X = t
  - LIVI = he?e V = r
  - …

- Plaintext: HereUpOnLeGrandAroseWithAGraveAndStatelyAirAnd BroughtMeTheBeetleFromAGlassCaseInWhichItWasEnclosedItW asABe

One-Time Pad (OTP)



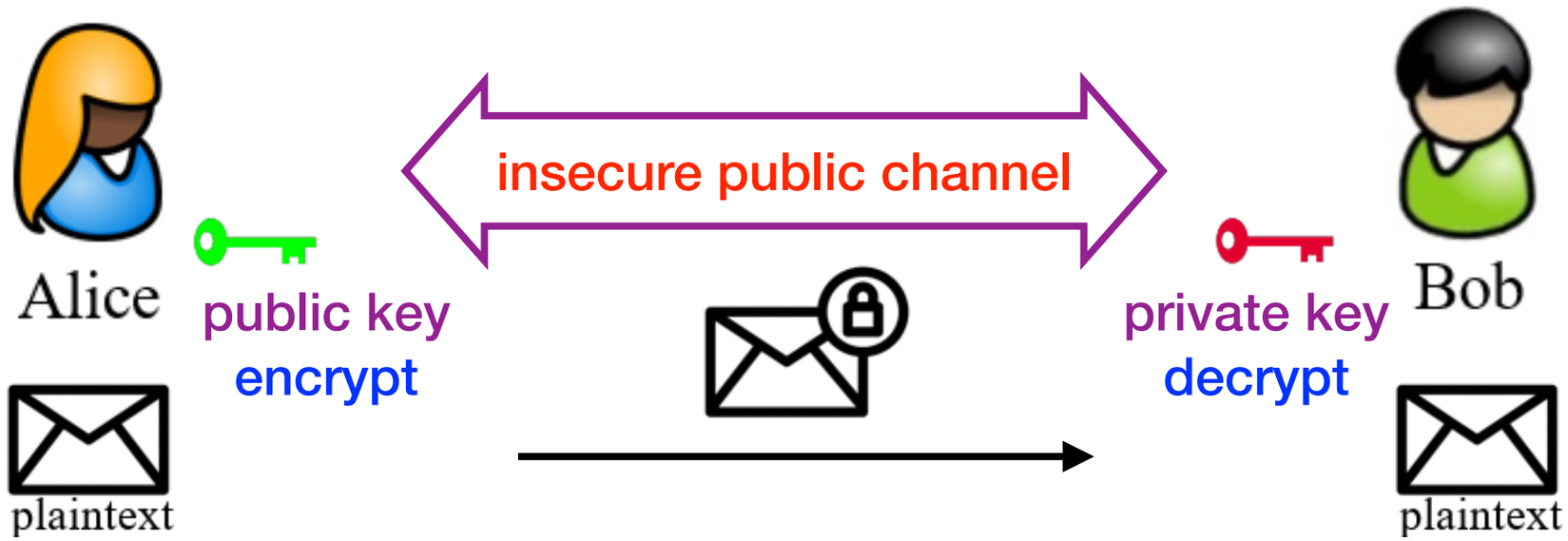
- Key (or pad): k = random binary string as long as the plaintext
- Encryption & Decryption: **xor** with the one-time pad (key) k
- Perfect secrecy: secure against even unlimited computing power
- End of story for symmetric cryptography?
  - ***No!*** OTP has very long random one-time keys, no integrity, etc.

** How to solve these issues? Take a cryptography course :)*

Asymmetric Cryptography

Asymmetric Cryptography

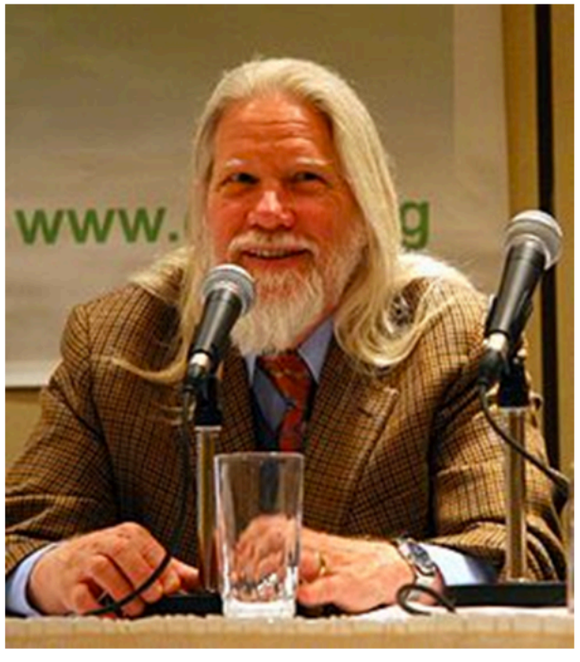
- **Asymmetric cryptography** focuses on cryptosystems where the message sender holds a public encryption key and the message receiver holds a private decryption key.
  - also known as **public-key cryptography**



Looks magic, right?

Public-Key Cryptography

- Public-key cryptography becomes real since the important breakthrough by **Diffie** and **Hellman** in 1976.
  - W. Diffie, M. Hellman, ***New directions in cryptography***, IEEE Transactions on Information Theory, vol. 22, pp. 644-654, 1976.
  - "We stand today on the brink of a revolution in cryptography."



Bailey W. Diffie

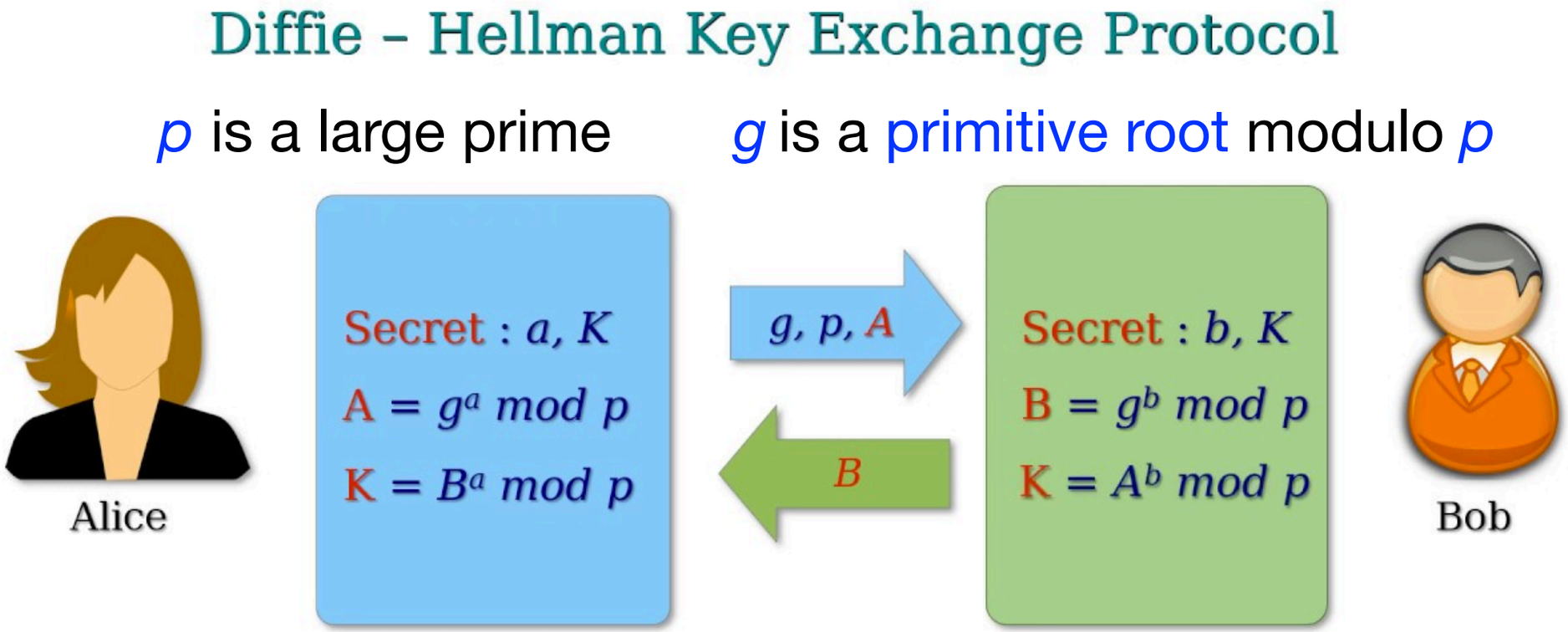
Martin E. Hellman

2015 Turing Award

for fundamental contributions to modern cryptography

Diffie-Hellman (DH) Key Exchange

- How to securely exchange keys (i.e., establish a shared secret) between two users over an insecure public channel?



Why is DH key exchange secure?

under modular arithmetic: *(a, b* sampled randomly from *{0, …, p – 2})*

knowing A and B does not help derive $K = A^b = B^a = g^{ab}$

The Discrete Logarithm Problem

- The discrete logarithm of an integer *y* to the base *b* modulo *m* is an integer *x* such that $b^x \equiv y \pmod m$.

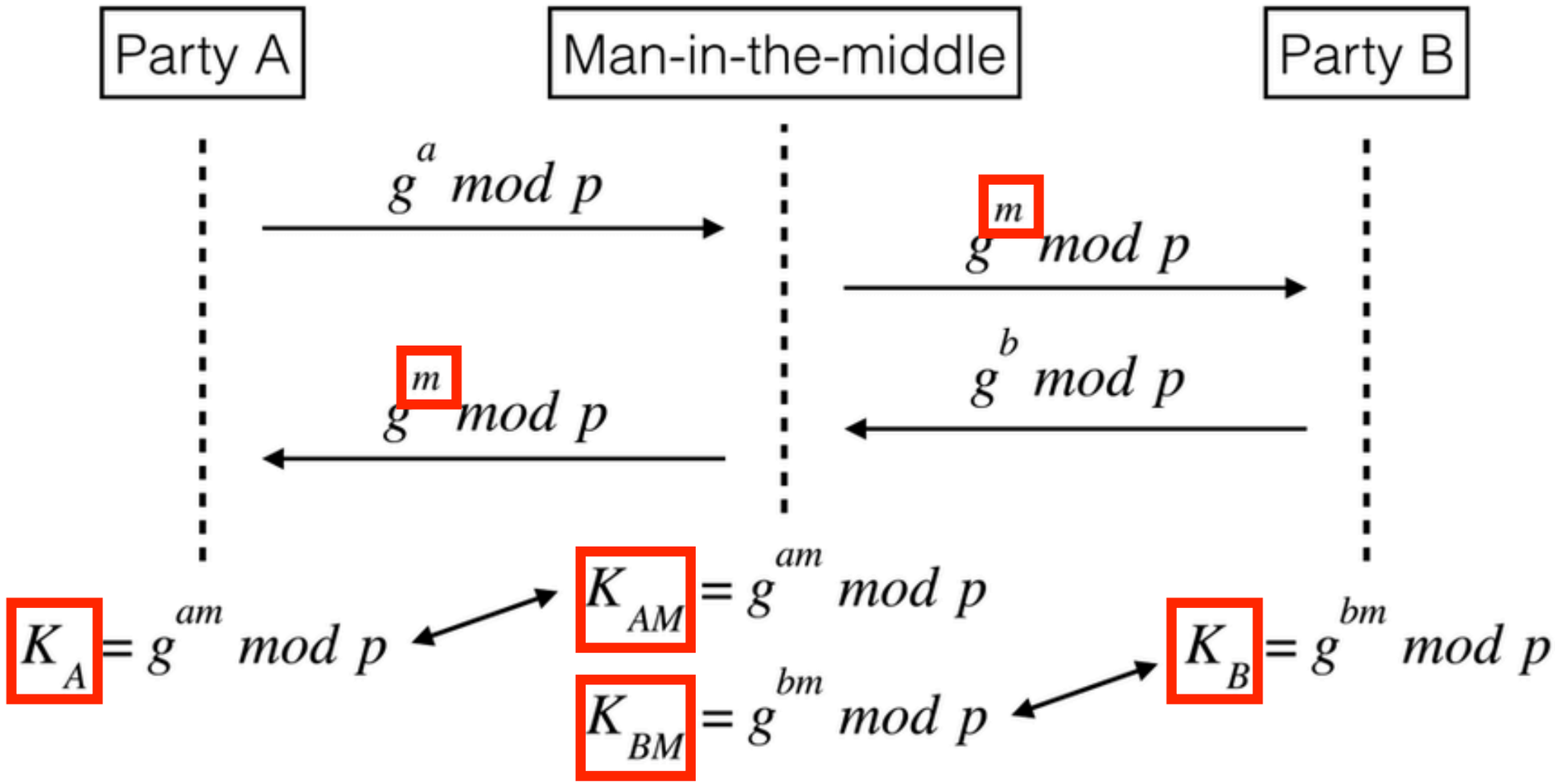
- **Discrete logarithm problem (DLP):** Given *m, b,* (random) *y,* find the discrete logarithm *x.*
  - In cryptography, usually *m* is a prime *p* and *b* is a primitive root *g*.
  - It is strongly believed that DLP is very hard and cannot be solved by any polynomial-time algorithms, i.e., DLP is not in class ***P***.
  - However, DLP can be efficiently solved using quantum computers.

- Security of DH key exchange is based on the hardness of DLP.
  - Actually based on stronger hardness assumptions (omitted here)
  - Actually ensures only security against passive attackers.

  ** What if attacks are active (also called man-in-the-middle attacks)?*

Man-In-The-Middle Attacks

- DH key exchange is insecure against man-in-the-middle attacks.



Wonder how this can be prevented? Take a cryptography course :)

SUSTech

The RSA Cryptosystem

- The widely used public-key cryptosystem **RSA** was invented by **R**ivest, **S**hamir and **A**dleman in 1977.
  - R. Rivest, A. Shamir, L. Adleman, ***A method for obtaining digital signatures and public-key cryptosystems***, Communications of the ACM, vol. 21-2, pages 120-126, 1978.

Ronald L. Rivest

Adi Shamir

Leonard M. Adleman

2002 Turing Award

for their ingenious contribution for making public-key cryptography useful in practice

RSA Encryption and Decryption

- Pick 2 large primes, p and q. Let $n = pq$, so $\phi(n) = (p - 1)(q - 1)$. The public encryption key (n, e) and private decryption key d are selected such that $gcd(e, \phi(n)) = 1$ and $ed \equiv 1 \pmod{\phi(n)}$.

- Encryption: encrypt m as $c = m^e \bmod n$ * (n, e) is the public key

- Decryption: decrypt c as $m = c^d \bmod n$ * d is the private key

- Correctness: For each integer $m \in \boldsymbol{Z}_n$ we have $m^{ed} \equiv m \pmod{n}$.
  - *the proof is left as an exercise for you (easy when gcd(m, n) = 1)*

- Security of RSA is based on the hardness of factoring n.
  - Actually based on stronger hardness assumption (omitted here)

- Note that, besides d, the values p, q, $\phi(n)$ must be kept secret!
  - E.g., finding $\phi(n)$ is equivalent to factoring $n = pq$

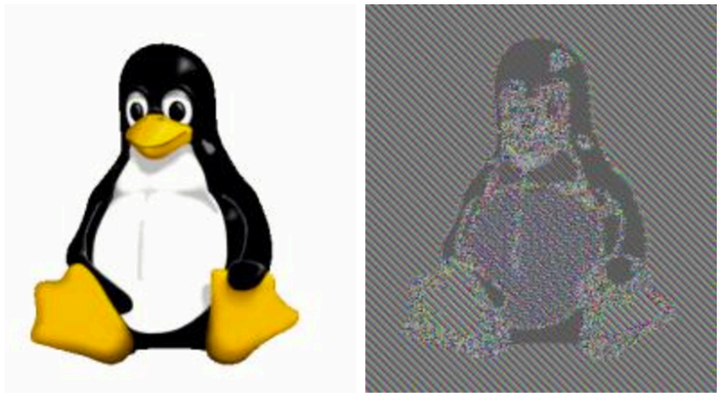
RSA Digital Signature

- Pick 2 large primes, p and q. Let $n = pq$, so $\phi(n) = (p - 1)(q - 1)$. The private signing key d and public verification key (n, e) are selected such that $gcd(e, \phi(n)) = 1$ and $ed \equiv 1 \pmod{\phi(n)}$.
- Signing: sign m as $s = m^d \bmod n$ * *d is the private key*
- Verification: verify (m, s) as $m = s^e \bmod n$ * *(n, e) is the public key*
- Correctness: For each integer $m \in \boldsymbol{Z}_n$ we have $m \equiv m^{de} \pmod{n}$.
  - same proof as with RSA encryption and decryption

- **essentially sign with decryption & verify with encryption**
  - Actually based on stronger hardness assumption (omitted here)
- Note that, besides d, the values p, q, $\phi(n)$ must be kept secret!
  - E.g., finding $\phi(n)$ is equivalent to factoring $n = pq$

Security of RSA

- In practice, RSA keys are typically 1024 to 2048 bits long and the random large primes p and q are sampled in a good way.
  - RSA can be broken using quantum computers (not yet available).
- The textbook/plain RSA (that we described) is not secure!
  - deterministic encryption: same ciphertext for same messages



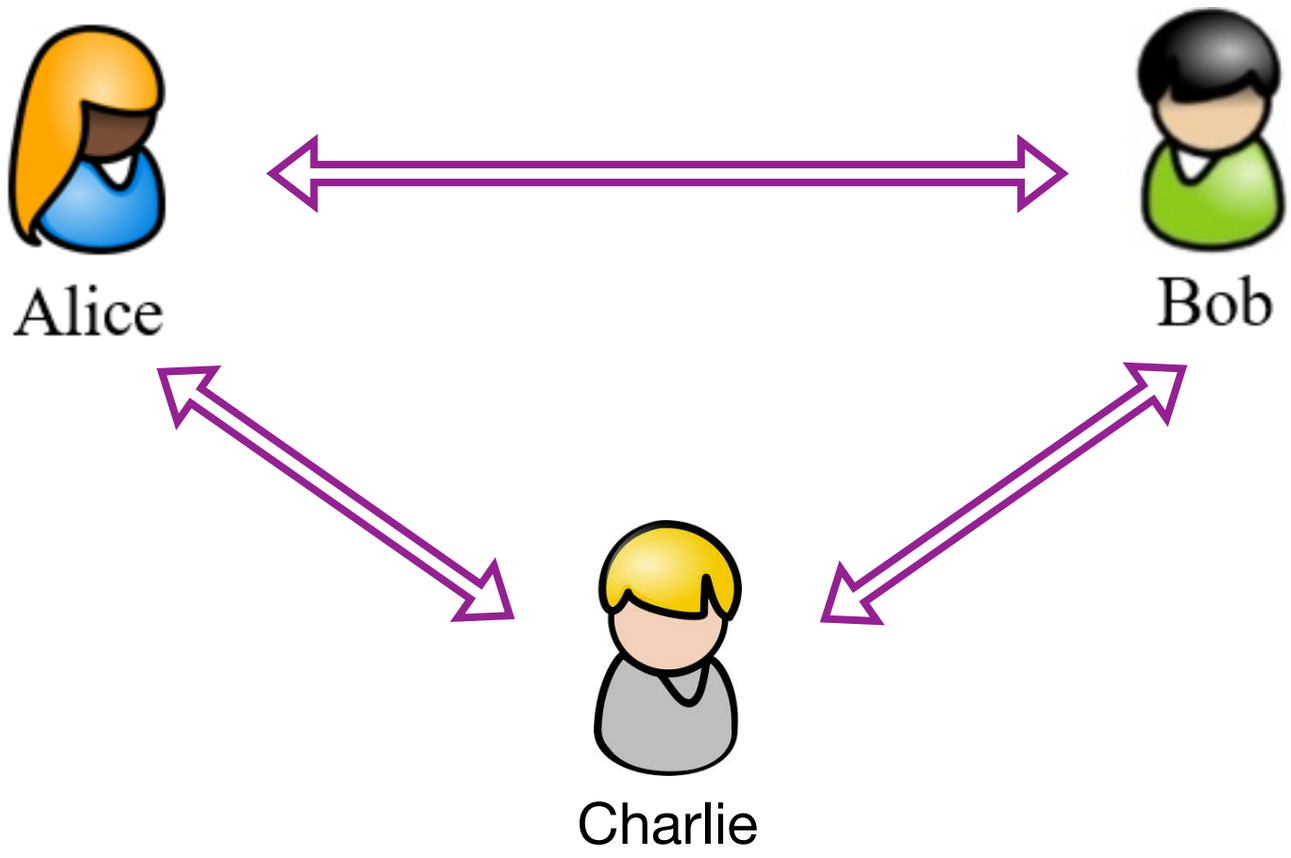
  - forgeable signatures: new signatures derived from existing ones
  - …

** Again, please take a cryptography course to know more :)*

Cryptographic Protocols

Cryptographic Protocols

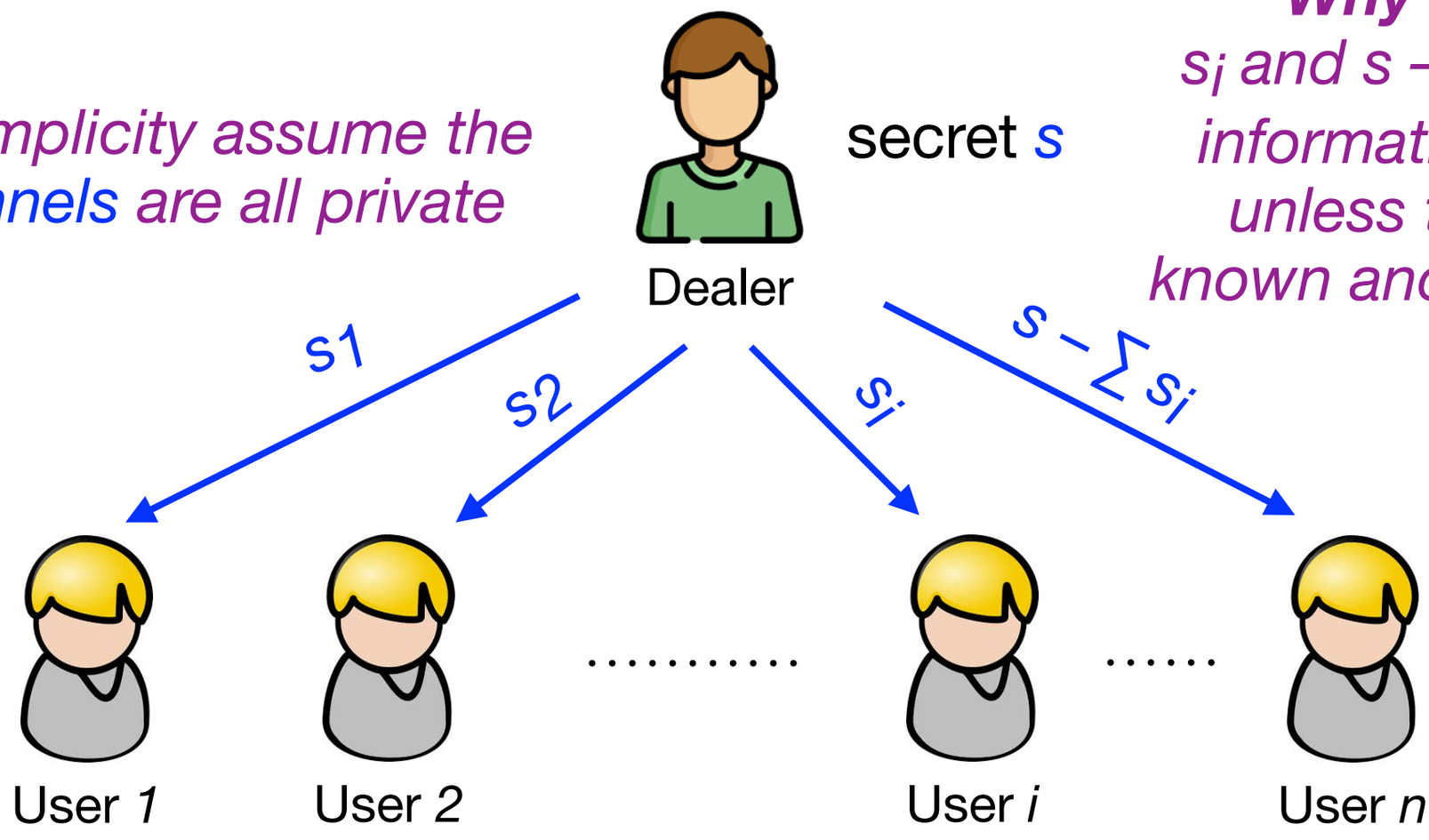
- A **cryptographic protocol** is an abstract or concrete protocol executed between multiple parties (users), which performs a security-related function and applies cryptographic methods, often as sequences of cryptographic primitives (building blocks).



(Trivial) Additive Secret Sharing

- How to securely share a secret s among multiple, say n, users?

for simplicity assume the channels are all private



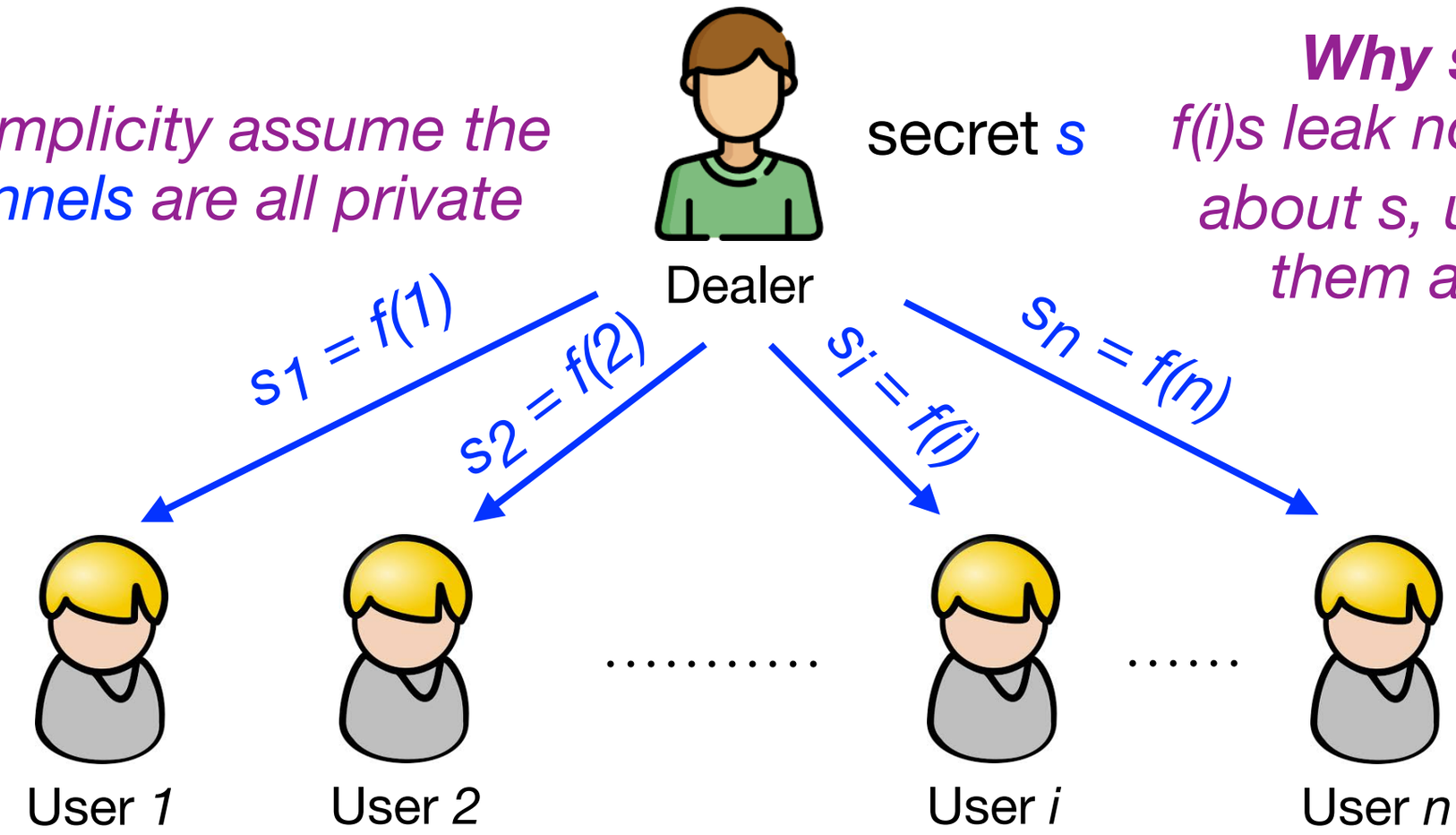
Why secure?
s_i and $s - \sum s_i$ leak no information about s, unless they are all known and summed up

To share a secret s, Dealer samples $n - 1$ random shares s_i, then distributes share s_i to user i and share $s - \sum s_i$ to user n.

Shamir's Threshold Secret Sharing

- Given a non-trivial threshold t $(1 < t < n)$, how to securely share a secret s, so that $\geq t$ users can reconstruct s but $< t$ users cannot?

for simplicity assume the channels are all private



Why secure?
f(i)s leak no information about s, unless ≥ t of them are known

To share a secret s, Dealer picks a random polynomial $f(n)$ with degree $\leq t - 1$ such that $f(0) = s$ (i.e., randomly sample coefficients $a_1, \ldots, a_{t-1}$ and let $a_0 = s$), then distributes share $s_i = f(i)$ to user i.

More Cryptography Topics

Symmetric Cryptography

- Encryption
- Stream ciphers
- Block ciphers
- Chosen plaintext attack
- Message integrity
- Message integrity from universal hashing
- Message integrity from collision resistant hashing
- Authenticated encryption
- …

Asymmetric Cryptography

- Public key tools
- Public key encryption
- Chosen ciphertext secure public key encryption
- Digital signatures
- Fast hash-based signatures
- Elliptic curve cryptography and pairings
- Attacks on number theoretic assumptions
- Post-quantum cryptography from lattices
- …

Cryptographic Protocols

- Protocols for identification and login
- Identification and signatures from Sigma protocols
- Proving properties in zero-knowledge
- Authenticated key exchange
- Threshold cryptography
- Secure multi-party computation
- …

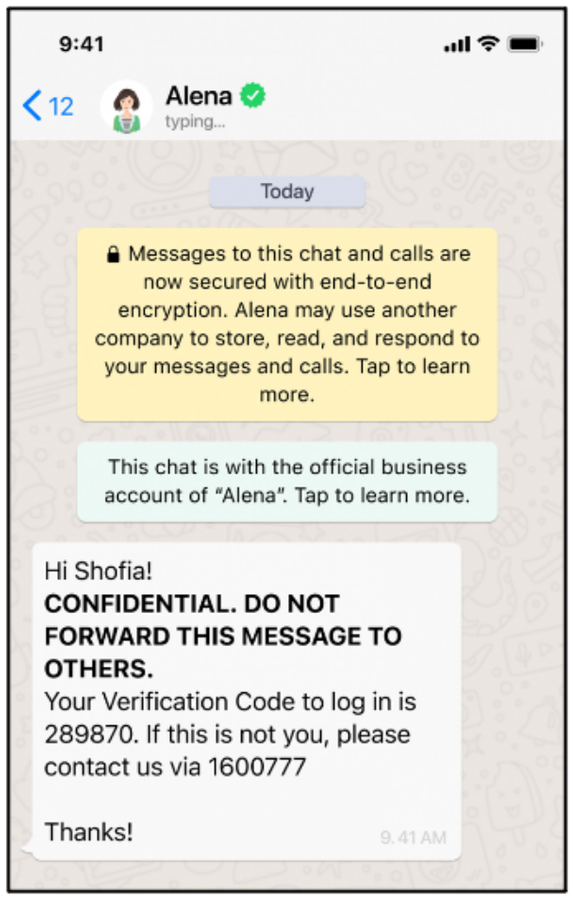
Real-World Cryptography

- Cryptography is widely used in the real world! ** my research area*

Secure Connections

Secure Authentication

Secure Messaging



Searchable Encryption

Blockchain Technology

Cryptography is a very cool and young area with lots of fascinating topics.
If you love math and CS theory, pls contact me to do crypto research!

SUSTech

06 Induction and Recursion

To be continued…

Midterm Exam and Assignment 3

- Midterm exam will take place in class (16:20~18:10) on Nov 1 and it captures materials from 02 Logic and Proofs to 05 Number Theory and Cryptography.
  - Midterm exam is closed-book.
  - Write your answers in English.
  - If your student ID **< 12311200**, please go to classroom **107**.
  - If your student ID **> 12311200**, please go to classroom **108**.
- Deadline for Assignment 3: Nov 8