National University
of computer and emerging sciences

DISCRETE STRUCTURES

COURSE INSTRUCTOR: MUHAMMAD SAIF UL ISLAM

Course Outline

- **Logic and Proofs** (Chapter 1)
- **Sets and Functions** (Chapter 2)
- **Relations** (Chapter 9)
- **Number Theory** (Chapter 4)
- Combinatorics and Recurrence
- Graphs
- Trees
- Discrete Probability

Lecture Outline

- ➢Divisibility and Modular Arithmetic
- ➢Primes and Greatest Common Divisors
- ➢Solving Congruencies
- ➢Applications of Congruencies
- ➢Cryptography
- ➢RSA

The part of mathematics devoted to the study of the set of integers and their properties is known as **number theory**.

Division

Definition: If *a* and *b* are integers with $a \neq 0$, then *a divides b* if there exists an integer *c* such that $b = ac$.

- When *a* divides *b* we say that *a* is a *factor* or *divisor* of *b* and that *b* is a multiple of *a*.
- The notation $a \mid b$ denotes that *a* divides *b*.
- If $a \mid b$, then b/a is an integer.
- If *a* does not divide *b*, we write $a \nmid b$.

Example: Determine whether $3 \mid 7$ and whether $3 \mid 12$.

$3 \nmid 7$, because 7/3 is not an integer.

On the other hand, $3 \mid 12$ because $12/3 = 4$ is an integer.

Properties of Divisibility

Theorem 1: Let a, b, and c be integers, where $a \neq 0$.

i. If $a \mid b$ and $a \mid c$, then $a \mid (b + c)$;

ii. If $a \mid b$, then $a \mid b.c$ for all integers c;

iii. If $a \mid b$ and $b \mid c$, then $a \mid c$.

Proof: (i) Suppose $a \mid b$ and $a \mid c$, then it follows that there are integers s and t with $b = as$ and $c = at$. Hence,

$b + c = as + at = a(s + t)$. Hence, $a \mid (b + c)$

(Exercises 3 and 4 ask for proofs of parts (ii) and (iii).)

Corollary: If a, b, and c be integers, where $a \neq 0$, such that $a \mid b$ and $a \mid c$, then $a \mid mb + nc$ whenever m and n are integers.

Can you show how it follows easily from from (ii) and (i) of Theorem 1?

Properties of Divisibility - Proof

If $a|b$ and $a|c$, then $a|(b + c)$;

statement	**reason**
$a,b,c \in Z$, $a|b$ and $a|c$	given
$b = a.q1$ and $c = a.q2$	def of division
such that $q1, q2 \in Z$	
$b+c = a.q1 + a.q2$	substitution
$b+c = a (q1 + q2)$	a as common
$q1+q2 = q3 \in Z$	closure property
$b+c = a.q3$	substitution
-> hence proved	

If $a|b$ and $b|c$, then $a|c$;

statement	**reason**
$a,b,c \in Z$, $a|b$ and $b|c$	given
$b = a.q1$ and $c = b.q2$	def of division
such that $q1, q2 \in Z$	
$c = b.q2$	substitution
$c = (a.q1).q2$	
$c = a (q1.q2)$	re-arranging
$q1.q2 = q3 \in Z$	closure property
$c = a.q3$	substitution
-> hence proved	

Division Algorithm

When an integer is divided by a positive integer, there is a quotient and a remainder. This is traditionally called the "Division Algorithm," but is really a theorem.

Theorem 2 (Division Algorithm): If a is an integer and d a positive integer, then there are unique integers q and r, with $0 \leq r < d$, such that $a = dq + r$ (*proved in Section* 5.2).

- d is called the *divisor*.
- a is called the *dividend*.
- q is called the *quotient*.
- r is called the *remainder*.

Definitions of Functions **div** and **mod**

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

Examples:

- What are the quotient and remainder when 101 is divided by 11?

  Solution: The quotient when 101 is divided by 11 is $9 = 101$ **div** 11, and the remainder is $2 = 101$ **mod** 11.
- What are the quotient and remainder when -11 is divided by 3?

  Solution: The quotient when -11 is divided by 3 is $-4 = -11$ **div** 3, and the remainder is $1 = -11$ **mod** 3.

Congruence Relation

Theorem 3:

Definition: If a and b are integers and m is a positive integer, then a is *congruent* to b *modulo* m if m divides $a - b$.

- The notation $a \equiv b \pmod{m}$ says that a is congruent to b modulo m.
- We say that $a \equiv b \pmod{m}$ is a *congruence* and that m is its *modulus.*
- Two integers are congruent mod m if and only if they have the same remainder when divided by m.
- If a is not congruent to b modulo m, we write

$$a \not\equiv b \pmod{m}$$

Example: Determine whether 17 is congruent to 5 modulo 6 and whether 24 and 14 are congruent modulo 6.

Solution:

- $17 \equiv 5 \pmod{6}$ because 6 divides $17 - 5 = 12$.
- $24 \not\equiv 14 \pmod{6}$ since 6 divides $24 - 14 = 10$ is not divisible by 6.

More on Congruences

Theorem 4: Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that $a = b + km$.

Proof:

- If $a \equiv b \pmod{m}$, then (by the definition of congruence) $m \mid a - b$. Hence, there is an integer k such that $a - b = km$ and equivalently $a = b + km$.
- Conversely, if there is an integer k such that $a = b + km$, then $km = a - b$. Hence, $m \mid a - b$ and $a \equiv b \pmod{m}$.

◀

The Relationship between (mod m) and **mod** m Notations

The use of "mod" in $a \equiv b \pmod{m}$ and a **mod** $m = b$ are different.

- $a \equiv b \pmod{m}$ is a relation on the set of integers.
- In a **mod** $m = b$, the notation **mod** denotes a function.

The relationship between these notations is made clear in this theorem.

Theorem: Let a and b be integers, and let m be a positive integer. Then $a \equiv b \pmod{m}$ if and only if a **mod** $m = b$ **mod** m. (*Proof in the exercises*)

$$a \textbf{ mod } m = b \textbf{ mod } m$$

Proof: a **mod** *m = b* **mod** *m*

Proof*: a* **mod** *m = b* **mod** *m*
assume a mod m= b mod m.

Let r= a mod m= b mod m.
Then, according to the division algorithm, there are q1, q2 ∈ Z such that
a= mq1+ r
b= mq2+ r, 0≤r<m.
Then a – b= mq1+ r – (mq2+ r)
= mq1+ r–mq2–r
= mq1–mq2
= m(q1–q2)

This shows that **m| (a–b)**, so **a ≡ b(mod m)**.We have proven that if **a mod m = b mod m** then a ≡ b (mod m).This concludes the proof.

Congruencies of Sums and Products

Theorem 5: Let m be a positive integer. If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then

$$a + c \equiv b + d \pmod{m} \text{ and } a.c \equiv b.d \pmod{m}$$

Example: Because $7 \equiv 2 \pmod{5}$ and $11 \equiv 1 \pmod{5}$, it follows from Theorem 5 that

$$\mathbf{18} = 7 + 11 \equiv 2 + 1 = \mathbf{3} \pmod{5}$$

$$\mathbf{77} = 7 \ . 11 \equiv 2 . 1 = \mathbf{2} \pmod{5}$$

Algebraic Manipulation of Congruencies

- Multiplying both sides of a valid congruence by an integer preserves validity.

  If $a \equiv b \pmod{m}$ holds then $c \cdot a \equiv c \cdot b \pmod{m}$, where c is any integer, holds by Theorem 5 with $d = c$.

- Adding an integer to both sides of a valid congruence preserves validity.

  If $a \equiv b \pmod{m}$ holds then $c + a \equiv c + b \pmod{m}$, where c is any integer, holds by Theorem 5 with $d = c$.

- Dividing a congruence by an integer does not always produce a valid congruence.

Example: The congruence $14 \equiv 8 \pmod{6}$ holds. But dividing both sides by 2 does not produce a valid congruence since $14/2 = 7$ and $8/2 = 4$, but $7 \not\equiv 4 \pmod{6}$.

Computing the **mod** *m* Function of Products and Sums

We use the following corollary to Theorem 5 to compute the remainder of the product or sum of two integers when divided by m from the remainders when each is divided by m.

Corollary: Let *m* be a positive integer and let *a* and *b* be integers. Then

$(a + b)\ (\mathbf{mod}\ m) = ((a\ \mathbf{mod}\ m) + (b\ \mathbf{mod}\ m))\ \mathbf{mod}\ m$

and

$ab\ \mathbf{mod}\ m = ((a\ \mathbf{mod}\ m)\ (b\ \mathbf{mod}\ m))\ \mathbf{mod}\ m.$

(*proof in text*)

Arithmetic Modulo *m*

Definitions: Let $\mathbf{Z}_m$ be the set of nonnegative integers less than *m*: $\{0,1, \ldots., m-1\}$

The operation $+_m$ is defined as $a +_m b = (a + b)$ **mod** *m*. This is *addition modulo m*.

The operation $\cdot_m$ is defined as $a \cdot_m b = (a \,.\, b)$ mod *m*. This is *multiplication modulo m*.

Using these operations is said to be doing *arithmetic modulo m*.

Example: Find $7 +_{11} 9$ and $7 \cdot_{11} 9$.

Solution: Using the definitions above:

- $7 +_{11} 9 = (7 + 9) \ \mathbf{mod}\ 11 = 16\ \mathbf{mod}\ 11 = 5$
- $7 \cdot_{11} 9 = (7 \cdot 9) \ \mathbf{mod}\ 11 = 63\ \mathbf{mod}\ 11 = 8$

Arithmetic Modulo *m*

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties as ordinary addition and multiplication.

- *Closure*: If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.
- ***Associativity***: If a, b, and c belong to $\mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$.
- ***Commutativity***: If a and b belong to $\mathbf{Z}_m$, then $a +_m b = b +_m a$ and $a \cdot_m b = b \cdot_m a$.
- ***Identity** elements*: The elements 0 and 1 are identity elements for addition and multiplication modulo m, respectively.
  - If a belongs to $\mathbf{Z}_m$, then $a +_m 0 = a$ and $a \cdot_m 1 = a$.

continued →

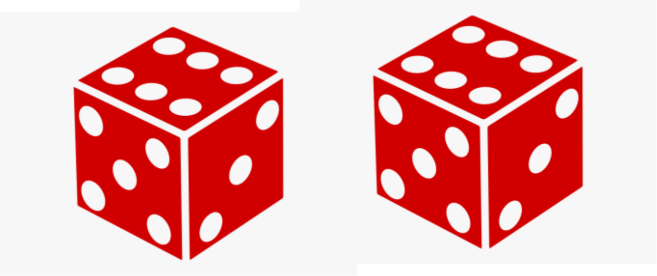
Arithmetic Modulo *m*

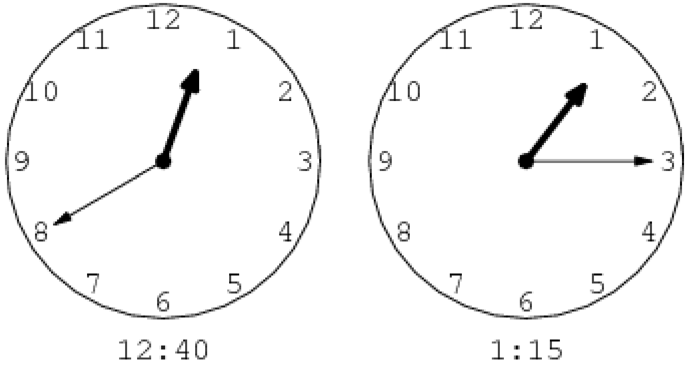
- ***Additive** inverses*: If $a \neq 0$ belongs to $\mathbf{Z}_m$, then $m - a$ is the additive inverse of a modulo m and 0 is its own additive inverse.
  - $a +_m (m - a) = 0$ and $0 +_m 0 = 0$
- ***Distributivity***: If *a*, *b*, and *c* belong to $\mathbf{Z}_m$, then
  - $a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$ and $(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c)$.

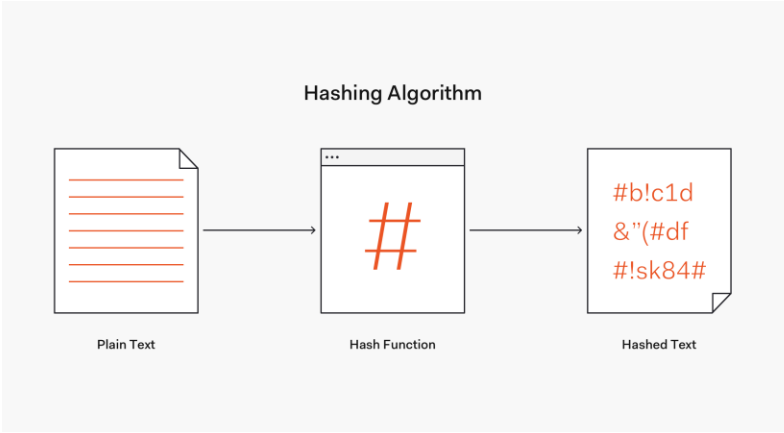
  - Note that we have listed the property that every element of **Z***m* has an additive inverse, but no analogous property for **multiplicative** inverses has been included. This is because multiplicative inverses do not always exist modulo *m*. For instance, there is no multiplicative inverse of 2 modulo 6

Exercises 42-44 ask for proofs of these properties.

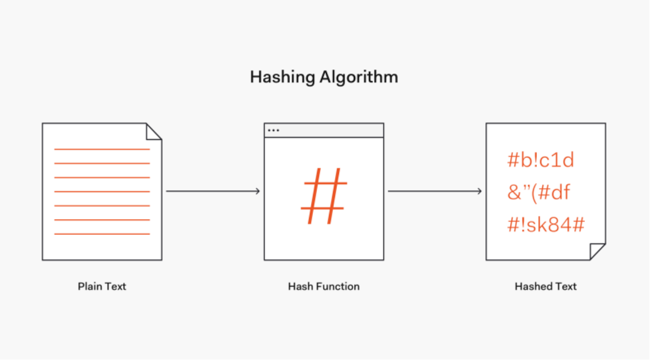
Applications of Congruences







Hashing Functions



Definition: A *hashing function h* assigns memory location *h*(*k*) to the record that has *k* as its key.

- A common hashing function is *h*(*k*) = *k* **mod** *m*, where *m* is the number of memory locations.
- Because this hashing function is onto, all memory locations are possible.

Example: Let *h*(*k*) = *k* **mod** 111. This hashing function assigns the records of customers with social security numbers as keys to memory locations in the following manner:

h(064212848) = 064212848 **mod** 111 = 14

h(037149212) = 037149212 **mod** 111 = 65

h(107405723) = 107405723 **mod** 111 = 14,

but since location 14 is already occupied, the record is assigned to the next available position, which is 15.

36 % 8 = 4

18 % 8 = 2

72 % 8 = 0

43 % 8 = 3

6 % 8 = 6

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
72		18	43	36		6	

The hashing function is not one-to-one as there are many more possible keys than memory locations. When more than one record is assigned to the same location, we say a *collision* occurs.

Here a collision has been resolved by assigning the record to the first free location.

Pseudorandom Numbers

Randomly chosen numbers are needed for many purposes, including computer simulations.

Pseudorandom numbers are not truly random since they are generated by systematic methods.

The *linear congruential method* is one commonly used procedure for generating pseudorandom numbers.

Four integers are needed: the *modulus m*, the *multiplier a*, the *increment c*, and *seed* x_0, with $2 \leq a < m, 0 \leq c < m, 0 \leq x_0 < m$.

We generate a sequence of pseudorandom numbers $\{x_n\}$, with $0 \leq x_n < m$ for all n, by successively using the recursively defined function

$$x_{n+1} = (ax_n + c) \textbf{ mod } m.$$

(*an example of a recursive definition, discussed in Section* 5.3*)*

If psudorandom numbers between 0 and 1 are needed, then the generated numbers are divided by the modulus, x_n /m.

Pseudorandom Numbers

Example: Find the sequence of pseudorandom numbers generated by the linear congruential method with modulus $m = 9$, multiplier $a = 7$, increment $c = 4$, and seed $x_0 = 3$.

Solution: Compute the terms of the sequence by successively using the congruence $x_{n+1} = (7x_n + 4)$ **mod** 9, with $x_0 = 3$.

$x_1 = 7x_0 + 4 \textbf{ mod } 9 = 7 \cdot 3 + 4 \textbf{ mod } 9 = 25 \textbf{ mod } 9 = 7,$

$x_2 = 7x_1 + 4 \textbf{ mod } 9 = 7 \cdot 7 + 4 \textbf{ mod } 9 = 53 \textbf{ mod } 9 = 8,$

$x_3 = 7x_2 + 4 \textbf{ mod } 9 = 7 \cdot 8 + 4 \textbf{ mod } 9 = 60 \textbf{ mod } 9 = 6,$

$x_4 = 7x_3 + 4 \textbf{ mod } 9 = 7 \cdot 6 + 4 \textbf{ mod } 9 = 46 \textbf{ mod } 9 = 1,$

$x_5 = 7x_4 + 4 \textbf{ mod } 9 = 7 \cdot 1 + 4 \textbf{ mod } 9 = 11 \textbf{ mod } 9 = 2,$

$x_6 = 7x_5 + 4 \textbf{ mod } 9 = 7 \cdot 2 + 4 \textbf{ mod } 9 = 18 \textbf{ mod } 9 = 0,$

$x_7 = 7x_6 + 4 \textbf{ mod } 9 = 7 \cdot 0 + 4 \textbf{ mod } 9 = 4 \textbf{ mod } 9 = 4,$

$x_8 = 7x_7 + 4 \textbf{ mod } 9 = 7 \cdot 4 + 4 \textbf{ mod } 9 = 32 \textbf{ mod } 9 = 5,$

$x_9 = 7x_8 + 4 \textbf{ mod } 9 = 7 \cdot 5 + 4 \textbf{ mod } 9 = 39 \textbf{ mod } 9 = 3.$

The sequence generated is 3,7,8,6,1,2,0,4,5,3,7,8,6,1,2,0,4,5,3,…

It repeats after generating 9 terms.

Commonly, computers use a linear congruential generator with increment $c = 0$. This is called a *pure multiplicative generator*. Such a generator with modulus $2^{31} - 1$ and multiplier $7^5 = 16{,}807$ generates $2^{31} - 2$ numbers before repeating.

Check Digits: UPCs

A common method of detecting errors in strings of digits is to add an extra digit at the end, which is evaluated using a function. If the final digit is not correct, then the string is assumed not to be correct.

Example: Retail products are identified by their *Universal Product Codes* (*UPC*s). Usually these have 12 decimal digits, the last one being the check digit. The check digit is determined by the congruence:

$$3x_1 + x_2 + 3x_3 + x_4 + 3x_5 + x_6 + 3x_7 + x_8 + 3x_9 + x_{10} + 3x_{11} + x_{12} \equiv 0 \pmod{10}.$$

a. Suppose that the first 11 digits of the UPC are 79357343104. What is the check digit?

b. Is 041331021641 a valid UPC?

Solution:

a. $3\cdot7 + 9 + 3\cdot3 + 5 + 3\cdot7 + 3 + 3\cdot4 + 3 + 3\cdot1 + 0 + 3\cdot4 + x_{12} \equiv 0 \pmod{10}$

$21 + 9 + 9 + 5 + 21 + 3 + 12 + 3 + 3 + 0 + 12 + x_{12} \equiv 0 \pmod{10}$

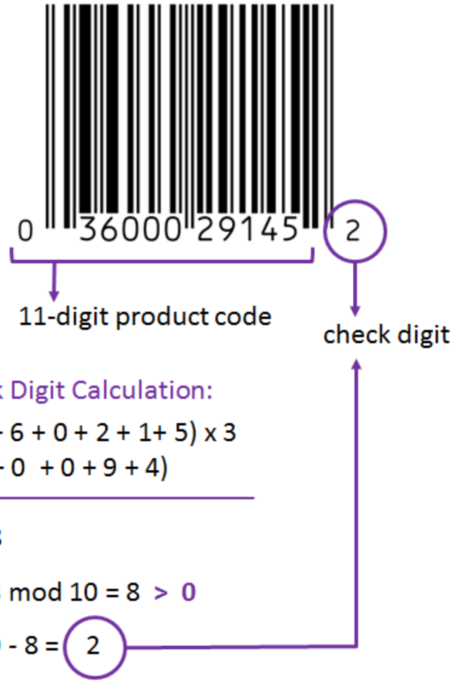
$98 + x_{12} \equiv 0 \pmod{10}$

$x_{12} \equiv 0 \pmod{10}$ So, the check digit is 2. (Needs to add 2 to make it 100)

b. $3\cdot0 + 4 + 3\cdot1 + 3 + 3\cdot3 + 1 + 3\cdot0 + 2 + 3\cdot1 + 6 + 3\cdot4 + 1 \equiv 0 \pmod{10}$

$0 + 4 + 3 + 3 + 9 + 1 + 0 + 2 + 3 + 6 + 12 + 1 = 44 \equiv 4 \not\equiv \pmod{10}$

Hence, 041331021641 is not a valid UPC.



Check Digit Calculation:

(0 + 6 + 0 + 2 + 1+ 5) x 3
+ (3 + 0 + 0 + 9 + 4)

= 58

58 mod 10 = 8 > 0

10 - 8 = 2

Check Digits: ISBNs

The **ISBN** is thirteen digits long if assigned on or after 1 January 2007, and **ten** digits long if assigned before 2007.

Books are identified by an *International Standard Book Number* (**ISBN-10**), a 10 digit code. The first 9 digits identify the language, the publisher, and the book. The tenth digit is a check digit, which is determined by the following congruence

$$x_{10} \equiv \sum_{i=1}^{9} i x_i \pmod{11}.$$

The validity of an ISBN-10 number can be evaluated with the equivalent

$$\sum_{i=1}^{10} i x_i \equiv 0 \pmod{11}.$$

a. Suppose that the first 9 digits of the **ISBN-10** are 007288008. What is the check digit?

b. Is 084930149X a valid ISBN10?

Solution:

a. $X_{10} \equiv 1 \cdot 0 + 2 \cdot 0 + 3 \cdot 7 + 4 \cdot 2 + 5 \cdot 8 + 6 \cdot 8 + 7 \cdot 0 + 8 \cdot 0 + 9 \cdot 8 \pmod{11}$.

$X_{10} \equiv 0 + 0 + 21 + 8 + 40 + 48 + 0 + 0 + 72 \pmod{11}$.

$X_{10} \equiv 189 \equiv 2 \pmod{11}$. Hence, $X_{10} = 2$.

b. $1 \cdot 0 + 2 \cdot 8 + 3 \cdot 4 + 4 \cdot 9 + 5 \cdot 3 + 6 \cdot 0 + 7 \cdot 1 + 8 \cdot 4 + 9 \cdot 9 + 10 \cdot 10 =$

$0 + 16 + 12 + 36 + 15 + 0 + 7 + 32 + 81 + 100 = 299 \equiv 2 \not\equiv 0 \pmod{11}$

Hence, 084930149X is not a valid ISBN-10.

X is used for the digit 10.

- A *single error* is an error in one digit of an identification number and a *transposition error* is the accidental interchanging of two digits. Both of these kinds of errors can be detected by the check digit for ISBN-10. (*see text for more details*)

Cryptography

Caesar Cipher

Julius Caesar created secret messages by shifting each letter three letters forward in the alphabet (sending the last three letters to the first three letters.) For example, the letter B is replaced by E and the letter X is replaced by A. This process of making a message secret is an example of ***encryption***.

Here is how the encryption process works:

- Replace each letter by an integer from $\mathbf{Z}_{26}$, that is an integer from 0 to 25 representing one less than its position in the alphabet.
- The encryption function is $f(p) = (p + 3)$ **mod** 26. It replaces each integer p in the set $\{0,1,2,\ldots,25\}$ by $f(p)$ in the set $\{0,1,2,\ldots,25\}$.
- Replace each integer p by the letter with the position $p + 1$ in the alphabet.

Example: Encrypt the message "MEET YOU IN THE PARK" using the Caesar cipher.

Solution: 12 4 4 19   24 14 20   8 13   19 7 4   15 0 17 10.

Now replace each of these numbers p by $f(p) = (p + 3)$ **mod** 26.

15 7 7 22   1 17 23   11 16   22 10 7   18 3 20 13.

Translating the numbers back to letters produces the encrypted message

"PHHW BRX LQ WKH SDUN."

Caesar Cipher

To recover the original message, use $f^{-1}(p) = (p - 3)$ **mod** 26. So, each letter in the coded message is shifted back three letters in the alphabet, with the first three letters sent to the last three letters. This process of recovering the original message from the encrypted message is called ***decryption***.

The Caesar cipher is one of a family of ciphers called *shift ciphers.* Letters can be shifted by an integer *k,* with 3 being just one possibility. The encryption function is

$$f(p) = (p + k) \textbf{ mod } 26$$

and the decryption function is

$$f^{-1}(p) = (p - k) \textbf{ mod } 26$$

The integer k is called a ***key***.

Shift Cipher

Example 1: Encrypt the message “STOP GLOBAL WARMING” using the shift cipher with $k = 11$.

Solution: Replace each letter with the corresponding element of $\mathbf{Z}_{26}$.

18 19 14 15   6 11 14 1 0 11   22 0 17 12 8 13 6.

Apply the shift $f(p) = (p + 11)$ **mod** 26, yielding

3 4 25 0   17 22 25 12 11 22   7 11 2 23 19 24 17.

Translating the numbers back to letters produces the ciphertext

“DEZA RWZMLW HLCXTYR.”

Shift Cipher

Example 2: Decrypt the message "LEWLYPLUJL PZ H NYLHA ALHJOLY" that was encrypted using the shift cipher with *k* = 7.

Solution: Replace each letter with the corresponding element of $\mathbf{Z}_{26}$.

$11\ 4\ 22\ 11\ 24\ 15\ 11\ 20\ 9\ 11\quad 15\ 25\quad 7\quad 13\ 24\ 11\ 7\ \ 0\quad 0\ 11\ 7\ \ 9\ \ 14\ \ 11\ \ 24.$

Shift each of the numbers by $-k = -7$ modulo 26, yielding

$4\ 23\ 15\ 4\ 17\ 8\ 4\ 13\ 2\ 4\quad 8\ 18\quad 0\quad 6\ 17\ 4\ \ 0\ \ 19\quad 19\ \ 4\ \ 0\ \ 2\ \ 7\ \ 4\ \ 17.$

Translating the numbers back to letters produces the decrypted message

"EXPERIENCE IS A GREAT TEACHER."

Cryptanalysis

CRYPTANALYSIS The process of recovering plaintext from ciphertext without knowledge of both the encryption method and the key is known as **cryptanalysis** or **breaking codes**. In general, cryptanalysis is a difficult process, especially when the encryption method is unknown.

Approach1: We can try to recover the message by shifting all characters of the ciphertext by each of the 26 possible shifts (including a shift of zero characters).

Approach2: The nine most common letters in English text and their approximate relative frequencies are E 13%, T 9%, A 8%, O 8%, I 7%, N 7%, S 7%, H 6%, and R 6%. To cryptanalyze ciphertext that we know was produced using a shift cipher, we decode the ciphertext by finding the relative frequencies of letters in the ciphertext.

Public Key Cryptography

- All classical ciphers, including shift ciphers and affine ciphers, are examples of **private key cryptosystems**. In a private key cryptosystem, once you know an encryption key, you can quickly find the decryption key.

- *E.g.* $c = (p + k)$ **mod** 26. can be decrypted using p = $(c − k)$ **mod** 26.

- To avoid the need for keys to be shared by every pair of parties that wish to communicate securely, in the 1970s cryptologists introduced the concept of **public key cryptosystems**.

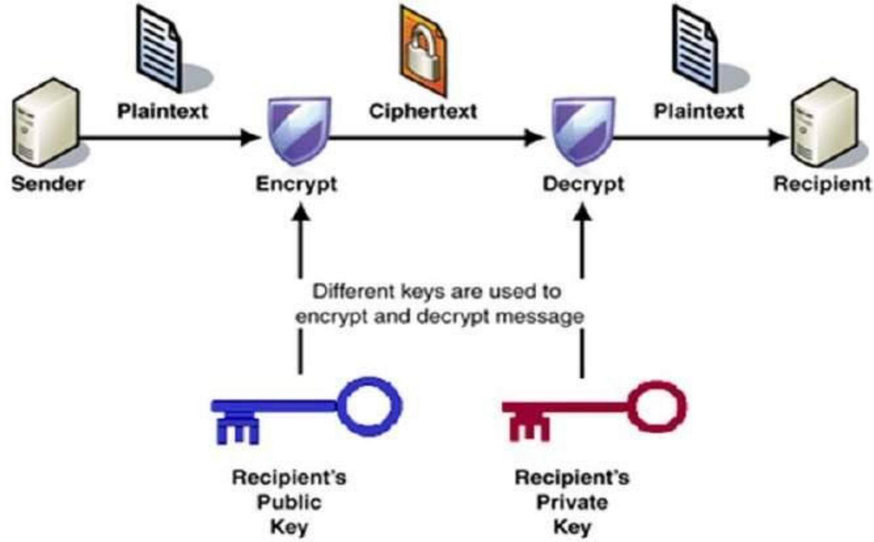
- Public-key cryptography, or asymmetric cryptography, is an encryption scheme that uses **two mathematically related**, but not identical, keys - a **public key** and a **private key**.

- Unlike symmetric key algorithms that rely on one key to both encrypt and decrypt, each key performs a unique function. The **public** key is used to **encrypt** and the **private** key is used to **decrypt**.

- PKC is also known as public key encryption, asymmetric encryption, asymmetric cryptography, asymmetric cipher, asymmetric key encryption and Diffie-Hellman encryption.

The RSA Algorithm

- Based on the idea that factorization of integers into their prime factors is hard.
- n=p ・ q, where p and q are distinct primes
- Proposed by Rivest, Shamir, and Adleman
- In 1977 and a paper was published in The Communications of ACM in 1978



Basic concept: https://youtu.be/AQDCe585Lnc

The RSA Algorithm

- Based on the idea that factorization of integers into their prime factors is hard.
- $n = p \cdot q$, where p and q are distinct primes
- Proposed by Rivest, Shamir, and Adleman in 1977 and a paper was published in The Communications of ACM in 1978
- Each individual has an encryption **key (*n, e*)**
- where ***n*** = *p.q*, the modulus is the product of two large primes *p* and *q*, say with 300 digits each,
- exponent ***e*** that is relatively prime to $(p - 1)(q - 1)$.

The RSA Encryption Algorithm

Step-1: To encrypt messages using a particular key (*n, e*), we first translate a plaintext message *M* into sequences of integers. To do this, we first translate each plaintext letter into a two-digit number, using the same translation we employed for shift ciphers.

Step-2: Next, we divide this string into equally sized blocks of 2*N* digits, where 2*N* is the largest even number such that the number 2525…25 with 2*N* digits does not exceed *n*. (When necessary, we pad the plaintext message with dummy Xs to make the last block the same size as all other blocks.)

Step-3: After these steps, we have translated the plaintext message *M* into a sequence of integers *m*1*, m*2*,…, mk* for some integer *k*. Encryption proceeds by transforming each block *mi* to a ciphertext block *ci*. This is done using the function $c = m^e$ **mod** *n.*

The RSA Encryption: Example

Note: For practical reasons we use small primes *p* and *q* in this example, rather than primes with 300 or more digits. Although the cipher described in this example is not secure, it does illustrate the techniques used in the RSA cipher.

Example: Encrypt the message **STOP** using the RSA cryptosystem with **key (2537, 13).** Note that $2537 = 43 \cdot 59$, $p = 43$ and $q = 59$ are primes, and $\gcd(e, (p-1)(q-1)) = \gcd(13, 42 \cdot 58) = 1$.

Solution: To encrypt, we first translate the letters in STOP into their numerical equivalents. We then group these numbers into blocks of four digits (because $2525 < 2537 < 252525$), to obtain 1819 1415.

We encrypt each block using the mapping $\boldsymbol{c = m^{13} \bmod 2537}$

Computations using fast modular multiplication show that 1819^{13} **mod** $2537 = 2081$ and

1415^{13} **mod** $2537 = 2182$. The encrypted message is **2081 2182.**

The RSA Decryption: Example

Example: We receive the encrypted message 0981 0461. What is the decrypted message if it was encrypted using the RSA cipher from Example 8?

Solution: The message was encrypted using the RSA cryptosystem with ***n* = 43 · 59** and exponent **13**. As Exercise 2 in Section 4.4 shows**, *d* = 937** is an inverse of **13 modulo 42 · 58 = 2436.** We use **937** as our decryption exponent. Consequently, to decrypt a block *c*, we compute

$$m = c^{937} \textbf{ mod } 2537.$$

To decrypt the message, we use the fast modular exponentiation algorithm to compute 0981^{937} **mod** 2537 = 0704 **and** 0461^{937} **mod** 2537 = 1115.

Consequently, the numerical version of the original message is 0704 1115

Translating this back to English letters, we see that the message is **HELP**

Example RSA

p = 7, q = 11, n = 77

Alice chooses e = 17, making d = 53

Bob wants to send Alice secret message

HELLO (07 04 11 11 14)

– 07^{17} mod 77 = 28; 04^{17} mod 77 = 16

– 11^{17} mod 77 = 44; – 11^{17} mod 77 = 44

– 14^{17} mod 77 = 42

- Bob sends **28 16 44 44 42**

Example RSA

Alice receives **28 16 44 44 42**

Alice uses private key, d = 53, to decrypt message:

– 28^{53} mod 77 = 07; 16^{53} mod 77 = 04

– 44^{53} mod 77 = 11; 44^{53} mod 77 = 11

– 42^{53} mod 77 = 14

- Alice translates **07 04 11 11 14** to ***HELLO***

No one else could read it, as only Alice knows her private key (needed for decryption)

Thank you!!!

Understanding Math by reading slides is similar to Learning to swim by watching TV.

So, DO PRACTICE IT!