

Lecture 5: Finite Fields (PART 2)

PART 2: Modular Arithmetic

Theoretical Underpinnings of Modern Cryptography

Lecture Notes on “Computer and Network Security”

by Avi Kak (kak@purdue.edu)

February 15, 2017
2:27am

©2017 Avinash Kak, Purdue University



Goals:

- To review modular arithmetic
- To present Euclid's GCD algorithms
- To present the prime finite field Z_p
- To show how Euclid's GCD algorithm can be extended to find multiplicative inverses
- **Perl and Python implementations for calculating GCD and multiplicative inverses**

CONTENTS

	<i>Section Title</i>	<i>Page</i>
5.1	Modular Arithmetic Notation	3
5.1.1	Examples of Congruences	5
5.2	Modular Arithmetic Operations	6
5.3	The Set Z_n and Its Properties	9
5.3.1	So What is Z_n ?	11
5.3.2	Asymmetries Between Modulo Addition and Modulo Multiplication Over Z_n	12
5.4	Euclid's Method for Finding the Greatest Common Divisor of Two Integers	15
5.4.1	Steps in a Recursive Invocation of Euclid's GCD Algorithm	17
5.4.2	An Example of Euclid's GCD Algorithm in Action	18
5.4.3	Proof of Euclid's GCD Algorithm	20
5.4.4	Implementing the GCD Algorithm in Perl and Python	21
5.5	Prime Finite Fields	28
5.5.1	What Happened to the Main Reason for Why Z_n Could Not be an Integral Domain	30
5.6	Finding Multiplicative Inverses for the Elements of Z_p	31
5.6.1	Proof of Bezout's Identity	33
5.6.2	Finding Multiplicative Inverses Using Bezout's Identity	36
5.6.3	Revisiting Euclid's Algorithm for the Calculation of GCD	38
5.6.4	What Conclusions Can We Draw From the Remainders?	40
5.6.5	Rewriting GCD Recursion in the Form of Derivations for the Remainders	41
5.6.6	Two Examples That Illustrate the Extended Euclid's Algorithm	43
5.7	The Extended Euclid's Algorithm in Perl and Python	44
5.8	Homework Problems	51

5.1: MODULAR ARITHMETIC NOTATION

- Given **any** integer a and a **positive** integer n , and given a division of a by n that leaves the remainder between 0 and $n - 1$, both inclusive, we define

$$a \bmod n$$

to be **the remainder**. Note that the remainder **must** be between 0 and $n - 1$, both ends inclusive, even if that means that we must use a negative quotient when dividing a by n .

- We will call two integers a and b to be **congruent modulo n** if

$$a \bmod n = b \bmod n$$

- Symbolically, we will express such a **congruence** by

$$a \equiv b \pmod{n}$$

- **Informally**, a congruence may also be displayed as:

$$a = b \pmod{n}$$

and even

$$a = b \pmod{n}$$

as long as it is understood that we are talking about a and b being equal only in the sense that their remainders obtained by subjecting them to modulo n division are exactly the same.

- We say a non-zero integer a is a **divisor** of another integer b provided the remainder is zero when we divide b by a . That is, when $b = ma$ for some integer m .
- When a is a divisor of b , we express this fact by $a \mid b$.

5.1.1: Examples of Congruences

- Here are some congruences modulo 3:

$$\begin{array}{rcl}
 7 & \equiv & 1 \pmod{3} \\
 -8 & \equiv & 1 \pmod{3} \\
 -2 & \equiv & 1 \pmod{3} \\
 7 & \equiv & -8 \pmod{3} \\
 -2 & \equiv & 7 \pmod{3}
 \end{array}$$

- One way of seeing the above congruences (for mod 3 arithmetic):

\dots 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 0 \dots
 \dots -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 11 12 \dots

where the top line is the output of **modulo 3** arithmetic and the bottom line the set of **all** integers. [The top entry in each column is the modulo 3 value of the bottom entry in the same column. Pause for a moment and think about the fact that whereas $(7 \pmod{3}) = 1$ on the positive side of the integers, on the negative side we have $(-7 \pmod{3}) = 2$.]

- As you can see, the **modulo n** arithmetic maps all integers into the set $\{0, 1, 2, 3, \dots, n - 1\}$.

[Click here to download full PDF material](#)