Introduction to Number Theory and Algorithms

Christian J. Rudder

August 2024

Contents

This page is left intentionally blank.

Big thanks to **Professor Leonid Levin**
for teaching CS235: Algebraic Algorithms
at Boston University.

Disclaimer: These notes are my personal understanding and interpretation of the course material. They are not officially endorsed by the instructor or the university. Please use them as a supplementary resource and refer to the official course materials for accurate information.

Preface

This is a Distillation of:
A Computational Introduction to Number Theory and Algebra
(Version 2), by Victor Shoup.

See https://shoup.net/ntb/ for the original text and practice problems.

Prerequisites

Definition 0.1: Well-Ordering Principle

Every non-empty set of positive integers has a least element.

Definition 0.2: "Without Loss of Generality"

A phrase that indicates that the proceeding logic also applies to the other cases. i.e., For a proposition not to lose the assumption that it works other ways as well.

— 1 —

Properties of Integers

1.1 Divisibility

a divides b, i.e., $\left(\frac{b}{a}\right)$, means b is reached by a, when a is multiplied by **some integer**.

Definition 1.1: Division

Let $a, b, x \in \mathbb{Z}$: $\left(\frac{b}{a}\right)$ means $b = ax$.

Denoted: $a|b$,
read a divides b, and a doesn't divide b is, $a \nmid b$.

Examples:

- $3 \mid 6$ because $6 = 3 \cdot 2$.
- $3 \nmid 5$ because $5 \neq 3 \cdot x$ for any $x \in \mathbb{Z}$.
- $2 \mid 0$ because $0 = 2 \cdot 0$.
- $0 \nmid 2$ because $2 \neq 0 \cdot x$ for any $x \in \mathbb{Z}$.

Note: $a, b, x \in \mathbb{Z}$ for, $\left(\frac{b}{a}\right)$ or $b = ax$ are labeled, a: divisor, b: dividend, x: quotient.

Tip: Many problems will involve manipulating equation like $b = ax$. Whether it's substituting b for ax or vice-versa, or adding/subtracting/multiplying/dividing.

Many definitions and theorems will build off one another. It's crucial to understand what concepts mean rather than memorizing them. This means having the ability to prove theorems and definitions from scratch.

Observe the following:

Theorem 1.1: Properties of Divisibility

For all $a, b, c \in \mathbb{Z}$:

(i) $a \mid a$, $1 \mid a$, and $a \mid 0$

(ii) $0 \mid a \iff a = 0$

(iii) $a \mid b \iff -a \mid b \iff a \mid -b$

(iv) $a \mid b \wedge a \mid c \implies a \mid (b + c)$

(v) $a \mid b \wedge b \mid c \implies a \mid c$

Try to prove these properties before reading the proof below.

Proof 1.1: Properties of Divisibility

Proof. For all $a, b, x, y \in \mathbb{Z}$:

(i)
- $a \mid a$ means $a = ax$, choosing $x = 1$ always satisfies.
- $1 \mid a$ because $a = 1 \cdot a$
- $a \mid 0$ because $0 = a \cdot 0$

(ii)
- If $0 \mid a$ then $a = 0 \cdot x$, 0 times any integer is 0, so $a = 0$
- If $a = 0$ then $0 = 0 \cdot x$, x can be any integer.

(iii) Proving $a \mid b \iff -a \mid b$:

- If $a \mid b$ then $b = ax = (-a)(-x)$, $-x$ is some integer, say x'.
  So $b = (-a)x'$ then $-a \mid b$
- If $-a \mid b$ then $b = (-a)x$, choose x to be some negative integer.

Proving $-a \mid b \iff a \mid -b$:

- If $-a \mid b$ then $b = (-a)x$, choose x positive integer.
- If $a \mid -b$ then $-b = ax$, choose x to be some negative integer.

(iv) If $a \mid b$ and $a \mid c$ then $b = ax$ and $c = ay$. Add both equations, $b + c = ax + ay$ factor, $b + c = a(x + y)$, $(x + y)$ is some integer. So $a \mid (b + c)$

(v) If $a \mid b$ and $b \mid c$ then $b = ax$ and $c = by$. Substitute b in c, $c = (ax)y$
shift terms, $c = a(xy)$, (xy) is some integer. So $a \mid c$.

■

Theorem 1.2: Reflexive Divisibility

For all $a, b \in \mathbb{Z}$: $a \mid b \wedge b \mid a \iff a = \pm b$. Additionally, $a \mid 1 \iff a = \pm 1$.

Proof 1.2: Reflexive Divisibility

Proof. For all $a, b, x, y \in \mathbb{Z}$:

Proving $a \mid b \wedge b \mid a \implies a = \pm b$:

$$\begin{aligned} a \mid b \quad b \mid a \qquad & \textit{Given} \\ b = ax \quad a = by \qquad & \textit{Definition of Division} \\ ab = (ax)(by) \qquad & \textit{Multiplying both equations} \\ ab = (ab)(xy) \qquad & \textit{Shift terms} \\ 1 = xy \qquad & \textit{Divide both sides by } ab \end{aligned}$$

x and y are integers, so $x = y = 1$. Substitute x and y,

$$\begin{aligned} b = a(1) \quad a = b(1) \qquad & \textit{Substitute} \\ a = b \qquad & \textit{Simplify} \end{aligned}$$

x or y could be $\pm$, so $\underline{a = \pm b.}$ Now $a = \pm b \implies a \mid b$ and $b \mid a$. From Theorem 1.2, we can use (i) to show $a \mid a$. Substitute b in for a, $\underline{a \mid b \text{ or } b \mid a.}$

Proving $a \mid 1 \implies a = \pm 1$:

$$\begin{aligned} a \mid 1 \qquad & \textit{Given} \\ 1 = ax \qquad & \textit{Definition of Division} \\ 1 = a(1) \qquad & \textit{Simplify} \end{aligned}$$

a must be 1, x could be $\pm$, so $a = \pm 1$ then $\underline{a \mid \pm 1 \text{ so } a \mid 1}$. ■

Definition 1.2: Cancellation Law

Let $a, b, c \in \mathbb{Z}$: If $ab = ac$ and $a \neq 0$ then $b = c$.

I.e., given $b = c$ multiplying both sides by a yields $ab = ac$, and still, $b = c$.

Definition 1.3: Prime Numbers

$p \in \mathbb{Z}$ is prime if $p \neq 0$ and p has no divisors other than 1 and p.

We will **only consider positive prime numbers**, in this text. Examples of primes are:

$$2, 3, 5, 7, 11, 13, 17, \ldots$$

Definition 1.4: Composite Numbers

$n, a, b \in \mathbb{Z}$ is composite if $n = ab$ and $1 < a < n$ and $1 < b < n$.

I.e., a composite number is a number that can be factor into two integers, other than 1 and itself.
Examples:

- 4 is composite because $4 = 2 \cdot 2$.
- 6 is composite because $6 = 2 \cdot 3$.

Briefly observe the following:

Theorem 1.3: Division Algorithm

For all $a, b \in \mathbb{Z}$, $b > 0$, there exists unique $q, r \in \mathbb{Z}$ such that $a = bq + r$ and $0 \leq r < |b|$.

To dissect, for all $a, b, q, r \in \mathbb{Z}$, $b > 0$, q and r exist uniquely such that:

$$a = bq + r$$

Dividend = **Divisor** · **Quotient** + **Remainder**

b fits into a q times with r left over.

Examples:

- $8 = 4 \cdot 2 + 0$
- $5 = 3 \cdot 1 + 2$

Note: Theorem 1.3 is called the Division Algorithm, despite not being an algorithm.

Proof 1.3: Division Algorithm

Proof. For all $a, b \in \mathbb{Z}$, $b > 0$, there exists unique $q, r \in \mathbb{Z}$ such that $a = bq + r$ and $0 \leq r < |b|$.

The definition of division $b \mid a$ then $a = bx$, $x \in \mathbb{Z}$. Subtract bx from both sides, $a - bx = 0$, working out evenly to 0. Freeze a and b, and vary x, yields a set of outputs, S:

$$S = \{a - bx : x \in \mathbb{Z}\}$$

"What's left of a after taking b, x times." E.g., $a = 6$, $b = 2$:

x	$a - bx$	
0	0	$= 6 - 2 \cdot 0$
1	4	$= 6 - 2 \cdot 1$
2	2	$= 6 - 2 \cdot 2$
3	0	$= 6 - 2 \cdot 3$
4	-2	$= 6 - 2 \cdot 4$

Let r be outputs of S and $q := x$ then $a - bq = r$ add bx to both sides, $\underline{a = bq + r}$.

Intuitively: I cut a cake of size a into pieces of b width for q people. Leftovers r can't exceed the size of the original cake: it's between nothing left or nothing shared, i.e., $\underline{0 \leq r < b}$.

We found our lower bound: $S = \{a - bx : x \in \mathbb{Z}, a - bx \geq 0\}$.

Formally: By the Well-Ordering Principle (0.1), there exists a smallest element in S, say r. To show S is not empty, choose $x = 0$ then $a - b(0) = a$, we are left with $0 \leq a$.

Without loss of generality, also assume $a < 0$. To satisfy $a - bx \geq 0$ choose $x = a$ yielding $a - ba = a(1 - b)$. Then $(1 - b) \leq 0$ as $(0 \leq r < b)$ so $(1 \leq b)$. Hence $a(1 - b) \geq 0$ as $(n < 0) \cdot (m \leq 0) = (h \geq 0)$ for some $n, m, h \in \mathbb{Z}$. So $\underline{S \text{ is not empty.}}$

• $\mathbf{r < b}$, say $r \geq b$, $\underline{r \text{ is the smallest element.}}$ Then $r = a - bq \geq b$. Subtract b from both sides, $(r - b = a - bq - b) \geq (b - b = 0)$ factoring we see $r - b = a - b(q + 1)$. Since $q + 1$ is some integer say q', $r - b = a - bq'$. There exists some b, $\underline{(r - b) < r \text{ contradicting our assumption.}}$

• $\mathbf{q, r}$ **uniqueness**, say there's another pair q', r' such that $a = (bq' + r') = (bq + r)$ and $0 \leq r' < b$. Without loss of generality, assume $r' \geq r$. Re-arrange both sides, $r' - r = bq - bq'$ factor, $r' - r = b(q - q')$. Then $b \mid (r' - r)$, but $(0 \leq r' - r < b)$ so $(r' - r) = 0$ therefore $r' = r$, showing $\underline{r \text{ is unique}}$. $b(q - q') = 0$ therefore $(q - q') = 0$ hence $q = q'$ showing $\underline{q \text{ is unique}}$. ■

1.2 Modular Arithmetic & Residues

Remember: For $a \in \mathbb{R}$, $a \in [0,1)$ is a range, i.e., including decimals from 0 to 1 (excluding 1).

Definition 2.1: Floor & Ceiling

For $x \in \mathbb{R}$ and $m, n \in \mathbb{Z}$. Functions map $\mathbb{R} \to \mathbb{Z}$,

Floor x, $\lfloor x \rfloor$, is the largest m such that $m \leq x < m + \varepsilon$, where $\varepsilon \in [0,1)$.
i.e., round down to the nearest integer.

Ceiling x, $\lceil x \rceil$, is the smallest n such that $n - \varepsilon < x \leq n$, where $\varepsilon \in [0,1)$.
i.e., round up to the nearest integer.

Definition 2.2: Mod Operator

Let $a, b \in \mathbb{Z}$, $b > 0$: The remainder of a divided by b. I.e., $a - b \left\lfloor \frac{a}{b} \right\rfloor$.

Denoted: "$a \bmod b$" or "$a \,\%\, b$".

Examples: $8 \bmod 3 = 2$, and $5 \bmod 2 = 1$

Proof 2.1: Mod Operator

The Division Algorithm (1.3) only works for $b > 0$. To generalize for $b < 0$,

$$\begin{aligned} a &= bq + r && \textit{Given} \\ a/b &= q + r/b && \textit{Divide both sides by } b \end{aligned}$$

We know $0 \leq r < b$, dividing b yielded $0 \leq \dfrac{r}{b} < 1$, so

$$\frac{r}{b} \in [0,1) \in \mathbb{R}$$

We notice $q = \left\lfloor \dfrac{a}{b} \right\rfloor$, as q is the largest integer that fits into a, b times. ■

Tip: $q = \left\lfloor \dfrac{a}{b} \right\rfloor$ is similar to integer division in programming, and $\dfrac{a}{b} = c$ implies $c \in \mathbb{R}$.

Theorem 2.1: Division Algorithm Extended

Let $a, b \in \mathbb{Z}$ with $b > 0$, and let $x \in \mathbb{R}$. Then there exist unique $q, r \in \mathbb{Z}$ such that $a = bq + r$ and $r \in [x, x + b)$.

$r \in [x, x + b)$ allows us to work with negative numbers and different intervals. Let's try to build some intuition about division and remainders:

$a, b, r \in \mathbb{Z}$ and $S = \{r = a - bq : q \in \mathbb{Z}\}$, $a = 6$, $b = 2$:

(0)

x	$a - bx$	
0	0	$= 6 - 2 \cdot 0$
1	4	$= 6 - 2 \cdot 1$
2	2	$= 6 - 2 \cdot 2$
3	0	$= 6 - 2 \cdot 3$
4	-2	$= 6 - 2 \cdot 4$
5	-4	$= 6 - 2 \cdot 5$
6	-6	$= 6 - 2 \cdot 6$
7	-8	$= 6 - 2 \cdot 7$

Dividing two numbers varying the divisor:

(1)

b	3 mod b
1	0
2	1
3	0
4	3
5	3
6	3
7	3
8	3

b	9 mod b
1	0
2	1
3	0
4	1
5	4
6	3
7	2
8	1
9	0
10	9

b	7 mod b
1	0
2	1
3	1
4	3
5	2
6	1
7	0
8	7

Grouping them by the remainder:

(2)

r	3 mod b
0	1, 3
1	2
3	4, 5, 6, …

r	9 mod b
0	1, 3, 9
1	2, 4, 8
3	5, 6, 7
9	10, 11, 12, …

r	7 mod b
0	1, 7
1	2, 6
2	5
3	4
7	8, 9, 10, …

Let's try the other way around.

(3)

a	a mod 3
0	0
1	1
2	2
3	0
4	1
5	2
6	0
7	1
8	2
9	0

a	a mod 9
0	0
1	1
2	2
3	3
…	…
9	0
10	1
11	2
…	…
18	0
19	1

a	a mod 7
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	0
8	1
9	2

Grouping them by the remainder:

(4)

r	a mod 3
0	0, 3, 6, 9
1	1, 4, 7
2	2, 5, 8

r	a mod 9
0	0, 9, 18
1	1, 10, 19
2	2, 11
3	3, 12
4	4, 13
5	5, 14
6	6, 15
7	7, 16
8	8, 17

r	a mod 7
0	0, 7
1	1, 8
2	2, 9
3	3
4	4
5	5
6	6

(5) **Table with increments of 3**

a	$a+1$	$a+2$
0	1	2
3	4	5
6	7	8
9	10	11
12	13	14
15	16	17
...	...	...

What is multiplication but repeated addition? What is division but repeated subtraction?

Column a in (5)-(7) shows multiples of b, which is example (4) transposed (highlighted). We can think of the width of a table as a's period.

Add 10 to 8, yields numbers always ending in 8. Add 5 to 8, yields numbers ending in 3 or 8. Then there are periods like (3).

We can see from the table (3), if we keep adding 3 to 2, we get 5, 8, 11, 14, etc.

(6) **Table with increments of 7**

a	$a+1$	$a+2$	$a+3$	$a+4$	$a+5$	$a+6$
0	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
...	...	...	...	...	...	...

(7) **Table with increments of 9**

a	$a+1$	$a+2$	$a+3$	$a+4$	$a+5$	$a+6$	$a+7$	$a+8$
0	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
...	...	...	...	...	...	...	...	...

We can represent these periods by $[x, x+b)$. Expanding the Division Algorithm (1.3) beyond $b > 0$, allows us to represent intervals no matter where we start on the number line.

We formally group (5)-(7)'s column headers into classes, which we call ***residues.***

Definition 2.3: Residue

Let $a, n \in \mathbb{Z}$, $n > 0$.

Set $R = \{a \bmod n : n \in \mathbb{Z}, n \neq 0\}$ produces remainders $r \in [0, n-1]$.
Each remainder r is a **residue** of a modulo n.

Definition 2.4: Residue Class

The set of numbers produced by a residue.

Denoted: $[a]_n$ or $a (\bmod n)$, a is the residue under modulo n.

Note: If modulo n is clear from context, then $[a]_n$, becomes $[a]$.

Definition 2.5: Representative

If $x \in [a]$, x is a representative of $[a]$.

1.3 Ring Theory

We will primarily focus on **ideals** and the behavior of primes; Though to understand ideals, is to understand **groups** , **rings**, and **fields**.

Definition 3.1: Group

A *group* is a set G that is closed under one operation, say '$*$', that satisfies four properties:

- **Closure:** For all $a, b \in G$, $a * b \in G$.
- **Associativity:** For all $a, b, c \in G$, $(a * b) * c = a * (b * c)$.
- **Identity:** There exists an element $e \in G$ such that for all $a \in G$, $a * e = e * a = a$.
- **Inverse:** $\forall a \, \exists \, a^{-1} \in G$, such that $a * a^{-1} = a^{-1} * a = e$ the identity.

Examples: The following are groups:

- Set $S = \{-1, 1\}$ closed under multiplication.
  - **Closure:** $-1 \cdot -1 = 1 \in S$.
  - **Associativity:** $(-1 \cdot 1) \cdot -1 = 1 \cdot -1 = -1$ and $-1 \cdot (1 \cdot -1) = -1 \cdot 1 = -1$.
  - **Identity:** 1, as $1 \cdot -1 = -1 \cdot 1 = -1$.
  - **Inverse:** -1, as $-1 \cdot -1 = 1 = 1$
- Set $I = \mathbb{Z}$ closed under addition.
  - **Closure:** $a + b \in I$ for all $a, b \in I$.
  - **Associativity:** $(a + b) + c = a + (b + c)$ for all $a, b, c \in I$.
  - **Identity:** 0, as $a + 0 = 0 + a = a$ for all $a \in I$.
  - **Inverse:** $-a$ for all $a \in I$, as $a + (-a) = (-a) + a = 0$.

Definition 3.2: Abelian Group

An *Abelian group* is a group that also satisfies the commutative property, i.e., for all $a, b \in G$, $a * b = b * a$. for some operation '$*$'.

Definition 3.3: Ring

A *ring* is a non-empty set R that is closed under additive ($+$) and multiplicative ($\cdot$) operations, such that:

- **Additive Structure:** (R) is an Abelian group.
- **Multiplicative Closure:** For all $a, b \in R$, $a \cdot b \in R$.
- **Distributive Property:** For all $a, b, c \in R$, $a \cdot (b + c) = a \cdot b + a \cdot c$ and $(a + b) \cdot c = a \cdot c + b \cdot c$.

Examples: $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, and $\mathbb{C}$ are all rings standard addition and multiplication.

Note: Operations aren't literally addition and multiplication. For example, the set of 2×2 matrices. Multiplication is different than standard applications, though exists in some form.

Tip: Numbers and symbols are just placeholders for the concepts they represent. 1,2 or ($\div$) don't have inherent properties; they are just symbols, changing meaning in different contexts.

Definition 3.4: Ideal

An *ideal* I, is a special subset of a ring R, such that for all $a, b \in I$ and $r \in R$:

- **Additive:** $a + b \in I$.
- **Multiplicative under the ring:** $a \cdot r \in I$ or $r \cdot a \in I$.
- **Additive identity:** There exists $e \in I : a + e = e + a = a$ (often 0).
- **Additive inverse:** $-a \in I : a + (-a) = -a + a = e$.

Example: The set of all multiples of 2, $2\mathbb{Z}$, is an ideal of $\mathbb{Z}$.

- **Additive:** $(2 \cdot a) + (2 \cdot b) = 2(a + b) \in 2\mathbb{Z}$.
- **Multiplicative:** $(2 \cdot a) \cdot r = 2(a \cdot r) \in 2\mathbb{Z}$.
- **Additive identity:** $0 \in 2\mathbb{Z}$.
- **Additive inverse:** For every $2a \in 2\mathbb{Z}$, its inverse is $-2a \in 2\mathbb{Z}$.

Definition 3.5: Field

A *field* is a ring $\mathbb{F}$ with additional properties:

- **Additive Structure:** $(\mathbb{F}, +)$ forms an Abelian group.
- **Multiplicative Structure:** $(\mathbb{F}, \cdot)$ forms an Abelian group excluding 0:
- **Distributive:** For all $a, b, c \in \mathbb{F}$, $a \cdot (b + c) = a \cdot b + a \cdot c$.

Example: $\mathbb{Q}$, the set of rational numbers:

- **Multiplicative identity:** $1 \in \mathbb{Q}$ as $1 \cdot a = a \cdot 1 = a$ for all $a \in \mathbb{Q}$.
- **Multiplicative inverse:** $a^{-1} = \frac{1}{a}$ as $a \cdot \frac{1}{a} = 1$.
- **Excludes 0:** As 0 has no multiplicative inverse, i.e., $\frac{1}{0}$ is undefined.

Tip: A **group** defines operations, an **abelian group** ensures commutativity, a **ring** R consists of an abelian group $(+)$, multiplication $(\cdot)$, and distribution. An **ideal** $I \subseteq R$ is a special subset of a ring, such that for $a \in I$ and $r \in R$, $a \cdot r \in I$. Finally, a **field** is a ring where the non-zero elements form a multiplicative abelian group.

1.4 Ideals & Generators

We will use $\mathbb{Z}$ as an ideal to explore the behavior of primes and divisibility.

Definition 4.1: Generator

An element or set of elements that can be used to *generate* a structure by repeated application of that structure's operations.

Denoted: $\langle a \rangle$, element a generating a structure.

Definition 4.2: Integer Ideal Generator

The ideal generated by an integer a in $\mathbb{Z}$, denoted $a\mathbb{Z}$, is the set of all multiples of a:

$$a\mathbb{Z} = \{a \cdot x : x \in \mathbb{Z}\} = \{\ldots, -2a, -a, 0, a, 2a, \ldots\}.$$

Example: The ideal generated by 2, $\langle 2 \rangle = \{\ldots, -4, -2, 0, 2, 4, \ldots\}$.

Proof 4.1: Proof that $a\mathbb{Z}$ is an Ideal

Let $a\mathbb{Z}$ be the ideal generated by $a \in \mathbb{Z}$, and let $az, az' \in a\mathbb{Z}$,$z'' \in \mathbb{Z}$, and $r \in \mathbb{R}$.

- **Additive Closure:** $az + az' = a(z + z') \in a\mathbb{Z}$.
- **Multiplicative Closure:** $az \cdot r = a(z \cdot r)$, then $(z \cdot r) \in \mathbb{Z}$ therefore $a(z \cdot r) \in a\mathbb{Z}$.
- **Additive Inverses:** $-az = a(-z) \in a\mathbb{Z}$.
- **Additive Identity:** $a \cdot 0 = 0 \in a\mathbb{Z}$.

Therefore, $a\mathbb{Z}$ is an ideal of $\mathbb{Z}$. ■

Definition 4.3: Principal Ideal

For ring R and $a \in R$, if $\langle a \rangle = \{r \cdot a : r \in R\}$ and $\langle a \rangle$ is an ideal of R, then $\langle a \rangle$ is a *principal ideal*.

Since $\mathbb{Z}$ forms a ring, for $a \in \mathbb{Z}$, $\langle a \rangle$ is a principal ideal of $\mathbb{Z}$. It also follows that $\langle a \rangle \subseteq \mathbb{Z}$.

Definition 4.4: Ideal Operations

Let I and J be ideals of a ring R.

- **Sum:** The sum of two ideals $I + J$ is defined as:

$$I + J = \{i + j : i \in I, j \in J\}.$$

  Since I and J are both have multiplicative closures of R, their sum is too.

$$(i \cdot r) \in I \text{ and } (j \cdot r) \in J \text{ then } (i \cdot r) + (j \cdot r) = (i + j) \cdot r \in I + J.$$

- **Product:** The product of two ideals $I \cdot J$ is defined as:

$$I \cdot J = \left\{ \sum i \cdot j : i \in I, j \in J \right\}.$$

  We need $\sum$ to show additive closure. We represent our product as sums alike $I + J$: For $i' \in I$:

$$(i \cdot j) + (i' \cdot j) = (i + i') \cdot j = i \cdot j \in I \cdot J.$$

Example: Consider ideals in $\mathbb{Z}$:

$$I = 2\mathbb{Z} = \{\ldots, -4, -2, 0, 2, 4, \ldots\} \quad \text{(the even integers)}$$

and

$$J = 3\mathbb{Z} = \{\ldots, -6, -3, 0, 3, 6, \ldots\} \quad \text{(the multiples of 3)}.$$

The product $I \cdot J$ is not just the set of all individual products like $2 \cdot 3 = 6$. Instead, it is the set of all sums of products of elements from I and J, including sums like:

$$2 \cdot 3 + (-2) \cdot 3 = 6 - 6 = 0$$

or

$$2 \cdot 3 + 4 \cdot 3 = 6 + 12 = 18.$$

Thus, the product of I and J is:

$$I \cdot J = \{\ldots, -18, -12, -6, 0, 6, 12, 18, \ldots\} = 6\mathbb{Z}.$$

Therefore, the product of $2\mathbb{Z}$ and $3\mathbb{Z}$ is $6\mathbb{Z}$, the set of multiples of 6. Illustrating $I \cdot J$ as the sums of products ensures the additive and multiplicative closure properties of ideals.

Theorem 4.1: Ideal Properties

For ideals in the integers $\mathbb{Z}$, and all $a, b \in \mathbb{Z}$:

- $b \in a\mathbb{Z}$ if and only if $a \mid b$.
- For every ideal $I \subseteq \mathbb{Z}$, $b \in I$ if and only if $b\mathbb{Z} \subseteq I$.
- Combining the above observations: $b\mathbb{Z} \subseteq a\mathbb{Z}$ if and only if $a \mid b$.

Proof 4.2: Proof of Ideal Properties

- $b \in a\mathbb{Z}$, let a be the smallest positive integer, then b must be 0, a, or some multiple of a, thus $a \mid b$. If $a \mid b$, then $b \in a\mathbb{Z}$, as $a\mathbb{Z}$ generates multiples of a.
- $b \in I$, then $b\mathbb{Z} \subseteq I$ as I upholds multiplicative closure. I.e., $q \in I$ then $bq \in I$.
- $a \mid b$, then $b \in a\mathbb{Z}$, $a\mathbb{Z}$ is an ideal, thus $b\mathbb{Z} \subseteq a\mathbb{Z}$. If $b\mathbb{Z} \subseteq a\mathbb{Z}$, then $b \in a\mathbb{Z}$, and $a \mid b$.

■

Theorem 4.2: Ideal Generator Existance of $\mathbb{Z}$

Let I be an ideal of $\mathbb{Z}$. Then there exists a unique non-negative integer d such that $I = d\mathbb{Z}$.

Proof 4.3: Proof of Generator Ideal equality of $\mathbb{Z}$

- **Existence:** $I = \{0\}$, then $d = 0$.
- $\mathbf{I \neq \{0\}}$: Let d be the smallest positive integer in I. If $a \in I$, then $d \mid a$, because $a = dq + r$ for some $q, r \in \mathbb{Z}$, where $0 \leq r < d$ (1.3). Since d is the smallest positive integer, $r = 0$, hence $d \mid a$.
- $\mathbf{I \subseteq d\mathbb{Z}}$, as $d \mid a$ and $a \in I$ (4.1).
- **Uniqueness:** Let d' be another non-negative integer. If $d'\mathbb{Z} = d\mathbb{Z}$, then $d \mid d'$ and $d' \mid d$. Thus, $d = \pm d'$ (1.2). Since, $d' \geq 0$, $d = d'$.

■

1.5 Primes & Greatest/Lowest Common Divisors

Definition 5.1: Greatest Common Divisor (GCD)

For all $a, b \in \mathbb{Z}$,
The *greatest common divisor* of a and b, is the largest positive integer dividing both a and b.
I.e., $d \in \mathbb{Z} : d \mid a$ and $d \mid b$, and d is unique.

Denoted: $\gcd(a, b)$.

Proof 5.1: GCD Existence and Uniqueness

Let $a, b \in \mathbb{Z}$, and $d = \gcd(a, b)$.

- **Existence:** d exists by the Well-Ordering Principle, as it's greatest element in the set of common divisors of a and b.
- **Uniqueness:** Let there be another GCD $d' \in \mathbb{Z}$ such that $d' \mid a$ and $d' \mid b$.
  Then, $d' \mid d$ and $d \mid d'$, so $d = \pm d'$ (1.2). GCD must be positive, so $d = d'$.

■

Theorem 5.1: GCD Ideal Linear Combination of $\mathbb{Z}$

For all $a, b, d \in \mathbb{Z}$ and $d = \gcd(a, b)$: $a\mathbb{Z} + b\mathbb{Z} = d\mathbb{Z}$

Proof 5.2: GCD Ideal Linear Combination of $\mathbb{Z}$

Let $I := a\mathbb{Z} + b\mathbb{Z}$. Then there exists $c \in Z$ such that $c\mathbb{Z} = I$ (4.2). Then $a, b, c \in I$, are all positive integers. We will prove facts of c:

- **Common Divisor:** $a, b \in I$ and $c\mathbb{Z} = I$. So $a, b \in c\mathbb{Z}$. Then $c \mid a$ and $c \mid b$ (4.1).
- **Linear Combination:** Since $c \in I$ and $a\mathbb{Z} + b\mathbb{Z} = I$. There exists some linear combination $as + bt = c$ for some $s, t \in \mathbb{Z}$ (4.4).
- **Greatest Divisor** Let $a, b \in I$ be the products $a = a'c'$ and $b = b'c'$, where $a', b' \in \mathbb{Z}$. Then there's a linear combination $a'c' + b'c' = c'(a' + b') = c$. So $c \mid c'$, hence c is the greatest common divisor of a and b.
- **Uniqueness:** By Lemma (5.1), c is unique, yielding $c = \gcd(a, b)$.

■

This next theorem heavily relies on Definition (4.1) and the previous Proof (5.1).

Theorem 5.2: Bezout's Identity

For all $a, b, d \in \mathbb{Z}$ and $d = \gcd(a, b)$: There exists some $s, t \in \mathbb{Z}$, such that $as + bt = r$ if and only if $d \mid r$. Moreover, if $d = 1$, then $as + bt = 1$.

Proof 5.3: Bezout's Identity

Let $r \in \mathbb{Z}$, and $d = \gcd(a, b)$, we have

$$\begin{aligned} as + bt = r &\iff r \in a\mathbb{Z} + b\mathbb{Z} && \text{(Ideal Multiplicative Closure (4.4))} \\ &\iff r \in d\mathbb{Z} && \text{(GCD Linear Combination (5.1))} \\ &\iff d \mid r && \text{(Property of Ideals (4.1))} \end{aligned}$$

■

Note: In $as + bt = r$, s and t are not unique, nor do they have to be positive: Example (4.4)

From above it follows that:

Definition 5.2: Reltively Prime

For all $a, b \in \mathbb{Z}$, a and b are *relatively prime* if $\gcd(a, b) = 1$.

Also known as a **coprime**.

I.e., given the equation $as + bt = r$ in (5.2), if $r = 1$, then a and b are coprime.

Examples:

- $\gcd(6, 9) = 3$, so 6 and 9 are not coprime.
- $\gcd(6, 7) = 1$, so 6 and 7 are coprime.

Tip: Étienne Bézout was a prominent 18th-century French mathematician, known for his contributions to algebra and number theory. He is most famous for **Bézout's Identity**. Bézout also contributed to algebraic geometry, notably with **Bézout's Theorem**, which gives the maximum number of intersections between two algebraic curves.

Theorem 5.3: Cancellation of GCD

Let $a, b, c \in \mathbb{Z}$ such that $c \mid ab$ and $\gcd(a, c) = 1$. Then $c \mid b$.

Proof 5.4: Coprime Coefficient Divisibility

Let $a, b, c \in \mathbb{Z}$ such that $c \mid ab$ and $\gcd(a, c) = 1$. a and c are coprime (5.2). Then there exists some $s, t \in \mathbb{Z}$ such that $as + ct = 1$ (5.2).Then,

$$\begin{aligned} as + ct &= 1 \text{ (Given)} \\ abs + cbt &= b \text{ (Multiply by } b\text{)} \\ cds + cbt &= b \text{ (Sub. } ab \text{ as } c \mid ab \Rightarrow ab = cd, d \in \mathbb{Z}\text{)} \\ c(ds + bt) &= b \text{ (Factor out } c\text{).} \end{aligned}$$

Yields $(ds + bt) \in \mathbb{Z}$, say m. So $cm = b$, hence $c \mid b$. ■

Theorem 5.4: Euclid's Lemma

Let p be a prime, and $a, b \in \mathbb{Z}$. If $p \mid ab$, then $p \mid a$ or $p \mid b$.

Proof 5.5: Euclid's Lemma

Let p be a prime, and $a, b \in \mathbb{Z}$ such that $p \mid ab$.

- If $p \mid a$, we satisfy the claim.
- If $p \nmid a$, then $\gcd(p, a) = 1$ (1.3). So by Cancellation of GCD (5.3), $p \mid b$.

■

Note: Primes only have two divisors: 1 and itself. So if $p \nmid a$, then $\gcd(p, a)$ must be 1.

Tip: Euclid is pronounced "You-clid". He was a Greek mathematician who lived around 300 BC. His work laid the foundation for number theory. He primarily worked on the properties of prime numbers, and is known for his algorithm to find the GCD of two numbers.

Our most important theorem, **The Fundamental Theorem of Arithmetic**:

Theorem 5.5: Fundamental Theorem of Arithmetic (FTA)

Every $n \in \mathbb{Z} : n > 1$ is prime or is product of primes, up to the order of the factors.

By "up to the order of the factors", we mean that the factorization is commutative.
Example: $30 = 2 \cdot 3 \cdot 5$ or $3 \cdot 2 \cdot 5$. The factorization is unique, except order (commutative).

Proof 5.6: Fundamental Theorem of Arithmetic

Let $n \in \mathbb{Z}$ be a non-zero integer.

Existence: by induction of n,

- **Base Case:** $n = 2$, which holds as 2 is prime.
- **Inductive Hypothesis:** Assume for all $n \leq k$, k is prime or product of primes.
- **Inductive Step:** Let $n = k + 1$.
  - If n is prime, then we're done.
  - n is not prime, then $n = ab \in \mathbb{Z}$ where $a \leq b < n$, otherwise $ab > n$. Reasoning: If $a \geq n$ or $b \geq n$, then $ab \geq n^2$, contradicting $ab = n$ unless $n = 1$, however $n \geq 2$.
  - **Recursively:** Then a and b are prime or product of primes (Inductive Hypothesis). Then n is a product of primes.

  Therefore by induction, every $n \in \mathbb{Z} : n > 1$ is prime or is product of primes.

Uniqueness: Let there be two different factorizations: $n = p_1 p_2 \dots p_k = q_1 q_2 \dots q_j$.
Both factorizations are products of primes. We divide both sides by p_1:

$$p_2 p_3 \dots p_k = \frac{q_1 q_2 \dots q_j}{p_1} \qquad (p_1)(p_2 \dots p_k) = (q_1 q_2 \dots q_j)$$

(Simplified left side) (Multiply by p_1)

Let $m := (p_1)$, $n := (p_2 \dots p_k)$, $k := (q_1 q_2 \dots q_j)$. Then $mn = k$, so $m \mid k$. Take out q_1 from k, then $m \mid q_1 \cdot k$. By Euclid's Lemma (5.5), $m \mid q_1$ or $m \mid k$ (the rest of the factors).

- If $m \mid q_1$, then $m = q_1$, by definition of prime (1.3)
- If $m \mid k$, then m equals some other prime in k.

Continuing from p_2 to p_k results in $p_i = q_i$ for all i, thus the factorization must be unique. ■

Theorem 5.6: Euclids Theorem

There are infinitely many primes.

Proof 5.7: Euclids Theorem

Say there are a finite number of primes: $p_1, p_2, \ldots, p_n$.
Let $M := p_1 \times p_2 \times \ldots \times p_n$ be the product of those primes. Let $N = M + 1$:

$$\begin{aligned} N &= M + 1 \\ N &= p_1 \times p_2 \times \ldots \times p_n + 1 \\ N &= (p_1)(p_2 \times \ldots \times p_n) + 1 \text{ (Form of Division Alg. (1.3))} \end{aligned}$$

N has remainder 1 when divided by any such prime. Thus, N is not a product of primes.
Then N must be a prime (5.5). ■

Extending the the Fundamental Theorem of Arithmetic (5.5):

Theorem 5.7: FTA Corollary

Every $n \in \mathbb{Z} : n > 1$ has a unique prime factorization, up to order and sign:

$$n = \pm p_1^{e_1} p_2^{e_2} \ldots p_k^{e_k}$$

For $p_1, p_2, \ldots, p_k$ distinct primes, and $e_1, e_2, \ldots, e_k$ positive integers.

Proof 5.8: FTA Corollary

Let $n \in \mathbb{Z} : n > 3$ and a composite number.

- $n = ab \in \mathbb{Z}$ by definition of a composite (1.4).
- Then a and b are prime or product of primes (5.5).
- Let a and b be prime and $a = b$

Then $n = a^2$, a prime squared. ■

We'll begin to define functions—which may or may not have logic—to abstract concepts.

Function 5.1: Prime Exponents - $\mathcal{V}_p(n)$

For each prime p where $n = p^e m \in \mathbb{Z}$ and $p \nmid m$. We define $\mathcal{V}_p(n) = e$.
I.e., $\mathcal{V}_p(n)$ is the exponent of p in the prime factorization of n.

In specifying $p \nmid m$, we ensure that p^e is the highest power of p dividing n.

We'll use this to abstract the Fundamental Theorem of Arithmetic further:

Theorem 5.8: FTA Abstracted by $\mathcal{V}_p(n)$

Every $n \in \mathbb{Z} : n > 1$ has a unique prime factorization, up to order and sign:

$$n = \pm \prod_p p^{\mathcal{V}_p(n)}$$

For p distinct primes.

Note: The notation $\prod$ is to products, as $\sum$ is to sums.

To expand our theorem for clarity:

$$n = \pm \prod_p p^{\mathcal{V}_p(n)} = \pm p_1^{\mathcal{V}_{p_1}(n)} p_2^{\mathcal{V}_{p_2}(n)} \cdots p_k^{\mathcal{V}_{p_k}(n)}$$

The "$\pm$" accounts for the sign of n. Say $n = -30$, then $n = -(2 \cdot 3 \cdot 5)$.

The function $\mathcal{V}_p(n)$ help us generalize GCD and **Least Common multiple** (LCM), but first we define two other functions:

Function 5.2: Minumum & Maximum - min(), max()

For all $a, b \in \mathbb{Z}$,

- $\min(a, b)$ is the smallest of a and b.
- $\max(a, b)$ is the largest of a and b.

Theorem 5.9: Operations of $\mathcal{V}_p(n)$

For all $a, b \in \mathbb{Z}$ and p prime:

- $\mathcal{V}_p(ab) = \mathcal{V}_p(a) + \mathcal{V}_p(b)$
- $\mathcal{V}_p(a^k) = k\mathcal{V}_p(a)$
- $a \mid b \iff \mathcal{V}_p(a) \leq \mathcal{V}_p(b)$ for all primes p

Proof 5.9: Operations of $\mathcal{V}_p(n)$

For all $a, b \in \mathbb{Z}$ and p prime:

- $\underline{\mathcal{V}_p(ab) = \mathcal{V}_p(a) + \mathcal{V}_p(b).}$
  Let $a = p^e m$ and $b = p^{e'} m'$, where $p \nmid m$ and $p \nmid m'$.
  - $ab = p^e m \times p^{e'} m' = p^{e+e'} mm'$
  - $\mathcal{V}_p(ab) = e + e' = \mathcal{V}_p(a) + \mathcal{V}_p(b)$

- $\underline{\mathcal{V}_p(a^k) = k\mathcal{V}_p(a).}$
  Let $a = p^e m$, where $p \nmid m$.
  - $a^k = (p^e m)^k = p^{ke} m^k$
  - $\mathcal{V}_p(a^k) = ke = k\mathcal{V}_p(a)$

- $\underline{a \mid b \iff \mathcal{V}_p(a) \leq \mathcal{V}_p(b) \text{ for all primes } p.}$
  Let $a = p^e m$ and $b = p^{e'} m'$, where $p \nmid m$ and $p \nmid m'$.
  - If $a \mid b$, then $b = aq \in \mathbb{Z}$. Thus, $\mathcal{V}_p(a) \leq \mathcal{V}_p(b)$, i.e., $e \leq e'$, otherwise $b < aq$.
  - $\mathcal{V}_p(a) \leq \mathcal{V}_p(b)$, both refer to p. Thus $a \mid b$ as a can pull some factor p^e out of b.

■

Tip: Remember that $\mathcal{V}_p(n)$ is some arbitrary function we defined. Despite this function being *made-up*, it has very **real** implications as we'll see in the next theorem. $\mathcal{V}_p(n)$ is no more real than $\gcd(a, b)$. It's a tool that helps us abstract.

Similar to how computers are built on binary logic, and then subsequently written in some abstracted higher-level language. Even those languages have their own abstractions through various libraries and frameworks, all helping speed up the process of development.

Theorem 5.10: GCD abstracted by $\mathcal{V}_p(n)$

For all $a, b \in \mathbb{Z}$:

$$\gcd(a,b) = \prod_p p^{\min(\mathcal{V}_p(a),\mathcal{V}_p(b))}$$

Proof 5.10: GCD abstracted by $\mathcal{V}_p(n)$

The $\gcd(a,b) = \prod_p p^{\min(\mathcal{V}_p(a),\mathcal{V}_p(b))}$ can be visualized into the following:

$a =$	$\prod$	$p_1^{e_1}$	$p_2^{e_2}$	$p_3^{e_3}$	$\dots$	$p_k^{e_k}$
$b =$	$\vdots$	$p_1^{e'_1}$	$p_2^{e'_2}$	$p_3^{e'_3}$	$\dots$	$p_k^{e'_k}$
$gcd(a,b) =$	$\vdots$	$p_1^{\min(e_1,e'_1)}$	$p_2^{\min(e_2,e'_2)}$	$p_3^{\min(e_3,e'_3)}$	$\dots$	$p_k^{\min(e_k,e'_k)}$

Separating a and b into their prime factors, taking the minimum exponent of each pair p_i, effectively intersecting all shared factors between a and b.

I.e., the GCD.

■

Definition 5.3: Least Common Multiple (LCM)

For all $a, b, n \in \mathbb{Z}$
The smallest positive integer divisible by both a and b is the *least common multiple*.
I.e., n is the smallest such integer that $a \mid n$ and $b \mid n$. n is unique.

Denoted: $\text{lcm}(a,b)$.

Proof 5.11: LCM Existence and Uniqueness

Let $a, b \in \mathbb{Z}$, and $m = \text{lcm}(a,b)$, m is the largest such integer that $a \mid m$ and $b \mid m$.

- **Existence:** m exists by the Well-Ordering Principle.
- **Uniqueness:** Let there be another smallest $m' \in \mathbb{Z} : a \mid m'$ and $b \mid m'$.
  Then, both m and m' are divisible by a and b. If m and m' are both the smallest, $m \mid m'$ and $m' \mid m$. Thus, $m = \pm m'$ (1.2). LCM must be positive, so $m = m'$.

■

Theorem 5.11: LCM abstracted by $\mathcal{V}_p(n)$

For all $a, b \in \mathbb{Z}$:

$$\operatorname{lcm}(a,b) = \prod_p p^{\max(\mathcal{V}_p(a),\mathcal{V}_p(b))}$$

Proof 5.12: LCM abstracted by $\mathcal{V}_p(n)$

The $\operatorname{lcm}(a,b) = \prod_p p^{\max(\mathcal{V}_p(a),\mathcal{V}_p(b))}$ can be visualized into the following:

$a =$	$\prod$	$p_1^{e_1}$	$p_2^{e_2}$	$p_3^{e_3}$	$\dots$	$p_k^{e_k}$
$b =$	$\vdots$	$p_1^{e'_1}$	$p_2^{e'_2}$	$p_3^{e'_3}$	$\dots$	$p_k^{e'_k}$
$lcm(a,b) =$	$\vdots$	$p_1^{\max(e_1,e'_1)}$	$p_2^{\max(e_2,e'_2)}$	$p_3^{\max(e_3,e'_3)}$	$\dots$	$p_k^{\max(e_k,e'_k)}$

Separating a and b into their prime factors, taking the maximum exponent of each pair p_i, effectively creating a union between a and b factors.

I.e., the LCM. ■

Theorem 5.12: GCD-LCM Relationship

For all $a, b \in \mathbb{Z}$:

$$\gcd(a,b) \cdot \operatorname{lcm}(a,b) = |ab|$$

which follows:

$$lcm(a,b) = \frac{|ab|}{\gcd(a,b)}$$

Proof on next page.

Proof 5.13: GCD-LCM Relationship

For every $a, b \in \mathbb{Z}$,
In Proof GCD abstracted (5.10) LCM abstracted (5.12) we showed:

- GCD creates the intersection of a and b's factors.
- LCM creates the union of a and b's factors.

LCM $\subseteq ab$'s factors. ab can be visualized as:

$a =$	$\prod$	$p_1^{e_1}$	$p_2^{e_2}$	$p_3^{e_3}$	$\dots$	$p_k^{e_k}$
$b =$	$\vdots$	$p_1^{e'_1}$	$p_2^{e'_2}$	$p_3^{e'_3}$	$\dots$	$p_k^{e'_k}$
$ab =$	$\vdots$	$p_1^{(e_1+e'_1)}$	$p_2^{(e_2+e'_2)}$	$p_3^{(e_3+e'_3)}$	$\dots$	$p_k^{(e_k+e'_k)}$

Then $\dfrac{|ab|}{\gcd(a,b)}$ would be:

$a =$	$\prod$	$p_1^{e_1}$	$\dots$	$p_k^{e_k}$
$b =$	$\vdots$	$p_1^{e'_1}$	$\dots$	$p_k^{e'_k}$
$lcm(a,b) =$	$\vdots$	$p_1^{(e_1+e'_1)-\min(e_1+e'_1)}$	$\dots$	$p_k^{(e_k+e'_k)-\min(e_k+e'_k)}$

We are done. To further illustrate, $\max(e_i + e'_i) = (e_i + e'_i) - \min(e_i + e'_i)$. Using $\mathcal{V}_p(n)$:

$$\operatorname{lcm}(a,b) = \prod_p p^{\mathcal{V}_p(ab) - \min(\mathcal{V}_p(a), \mathcal{V}_p(b))}$$

■

Note: $\mathcal{V}_p(ab) = \mathcal{V}_p(a) + \mathcal{V}_p(b)$ as shown in Theorem (5.9).

— 2 —

Congruences

2.1 Equivalence Relations

Definition 1.1: Equivalence Relation

An **equivalence relation** on set S is a relation $\sim$ which satisfies:

1. **Reflexivity:** For all $a \in S$, $a \sim a$.
2. **Symmetry:** For all $a, b \in S$, if $a \sim b$, then $b \sim a$.
3. **Transitivity:** For all $a, b, c \in S$, if $a \sim b$ and $b \sim c$, then $a \sim c$.

With $a \sim a$ reading, "a is related to a."

Definition 1.2: Equivalence Class

For $\sim$ equivalence relation on set S. For each $a \in S$, the **equivalence class** of a is the set

$$[a] = \{x \in S \mid x \sim a\}.$$

Note: For $x \in [a]$, x is a **representative** of the equivalence class $[a]$ (2.5).

Theorem 1.1: Equivalence Class Uniqueness

For $\sim$ equivalence relation on set S, for all $a, b \in S$:

(i) $a \in [a]$.

(ii) $a \in [b] \implies [a] = [b]$.

Proof 1.1: Equivalence Class Uniqueness

For $a, b \in S$:

(i) Since $\sim$ is reflexive, $a \sim a$.

(ii) Suppose $a \in [b]$. Then $a \sim b$. Then for $x \in S$,

$$\begin{aligned} x \in [a] &\implies x \sim a \text{ (Definition of } [a] \text{ (1.2))} \\ &\implies x \sim b \text{ (Transitivity, } x \sim a \wedge a \sim b) \\ &\implies x \in [b] \text{ (Definition of } [b] \text{ (1.2))} \end{aligned}$$

Thus $[a] \subseteq [b]$. Similarly, $[b] \subseteq [a]$. Therefore $[a] = [b]$.

■

2.2 Modular Congruences

Continuing with the notion of residues in, we introduce the concept of modular congruences (2.3).

Definition 2.1: Modular Congruence

For $n \in \mathbb{N}$, $a, b \in \mathbb{Z}$, a is **congruent** to b modulo n if $n \mid (a - b)$, denoted as

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

If $n \nmid (a - b)$, then $a \not\equiv b \pmod{n}$.

I.e., a and b have the same remainder when divided by n.

Note: $a \equiv b \pmod{n}$: **a** and **b** are **dividends** of **n** our **divisor**, which relate by **remainder**.

Theorem 2.1: Modular Congruence Properties

For all $a, b, c \in \mathbb{Z}$, and some positive integer n:

(i) $a \equiv a \pmod{n}$;

(ii) $a \equiv b \pmod{n} \implies b \equiv a \pmod{n}$;

(iii) $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n} \implies a \equiv c \pmod{n}$.

Proof 2.1: Modular Congruence Properties

For all $a, b, c \in \mathbb{Z}$, and some positive integer n:

(i) $a \equiv a \pmod{n}$ so $n \mid (a - a)$, which holds.

(ii) $a \equiv b \pmod{n}$ so n divides $(a - b)$ and $-(a - b) = (b - a)$, then $b \equiv a \pmod{n}$.

(iii) $a \equiv b \pmod{n}$ so $n \mid (a - b)$, and $b \equiv c \pmod{n}$ is $n \mid (b - c)$. Therefore,

$$\begin{aligned} & n \mid (a-b) \quad \text{and} \quad n \mid (b-c) \\ \implies \quad & n \mid [(a-b)+(b-c)] \\ \implies \quad & n \mid (a-c) \\ \implies \quad & a \equiv c \pmod{n}. \end{aligned}$$

■

Theorem 2.2: Modular Arithmetic

Let $a, a', b, b', n \in \mathbb{Z}$ with $n > 0$. If

$$a \equiv a' \pmod{n} \quad \text{and} \quad b \equiv b' \pmod{n},$$

then

$$a + b \equiv a' + b' \pmod{n} \quad \text{and} \quad a \cdot b \equiv a' \cdot b' \pmod{n}.$$

Proof 2.2: Modular Arithmatic

Addition: For $a, a', b, b', n \in \mathbb{Z}$,

- So $a \equiv a' \pmod{n}$ then $n \mid (a - a')$ means $a - a' = nx$ for some $x \in \mathbb{Z}$.
- Similarly, $b \equiv b' \pmod{n}$ then $b - b' = ny$ for some $y \in \mathbb{Z}$.
- Adding both equations, $(a - a') + (b - b') = (nx + ny)$ so $(a + b) - (a + b') = n(x + y)$.
- Therefore, $a + b \equiv a' + b' \pmod{n}$, as $n \mid (a + b) - (a' + b')$.

Multiplication: Continuing,

- If we multiply both equations, $(a - a')(b - b') = (nx)(ny)$ so $(ab) - (a'b') = n(xy)$.
- Therefore, $ab \equiv a'b' \pmod{n}$, as $n \mid (ab) - (a'b')$.

■

Theorem 2.3: Least Residue

Let $a, n \in \mathbb{Z}$ with $n > 0$. There exists unique $z \in \mathbb{Z}$ such that:

(i) $0 \leq z < n$,

(ii) $a \equiv z \pmod{n}$.

(iii) z is the **least residue** of a modulo n.

Particularly, for all $x \in \mathbb{Z}$, $z \in [x, x + n)$.

I.e., the least non-negative remainder r, which could be thought of as $r := a \bmod n$.

Note: The period $[x, x + n)$, contains possible remainders, a call back to the Division Alg. (2.1).

Proof 2.3: Least Residue

For some $a, q, n, r \in \mathbb{Z}$,
The Division Algorithm grantees existence, for $a = qn + r : 0 \leq r < n$ (2.1). Residues mod $n > 0$ are non-empty. Thus by the Well-Ordering Principle, there's a least element. ■

Example: Working to find the **set of solutions z** for $a \equiv z \pmod{n}$, i.e., find z that satisfies,

$$\begin{aligned} 3z + 4 &\equiv 6 \pmod{7} \text{ (Given)} \\ 3z &\equiv 2 \pmod{7} \text{ (Subtracting 4 from both sides)} \end{aligned}$$

We can't necessarily divide, but we can shift residue by some favorable factor.

$$\begin{aligned} 3z \cdot 5 &\equiv 2 \cdot 5 \pmod{7} \text{ (Multiply 5 to both sides)} \\ 1 \cdot z &\equiv 10 \pmod{7} \text{ (Since } 15 \equiv 1 \pmod{7}) \end{aligned}$$

Finding solution $z \equiv 10 \pmod{7}$, which we can reduce to $z \equiv 3 \pmod{7}$, as $3 \equiv 10 \pmod{7}$.
We say "integers z has solutions" as $z \in [3]_7 = \{3 + 7k : k \in \mathbb{Z}\}$ possible solutions.

Note: $[3]_7$ reads as *"the residue class 3 modulo 7."* Mentioned in (1.2).

2.3 Solving Linear Congruences

Theorem 3.1: Modular Multiplicative Identities

Let $a, n \in \mathbb{Z}$ with $n > 0$, and let $d := \gcd(a, n)$.

(i) For every $b \in \mathbb{Z}$, the congruence $az \equiv b \pmod{n}$ has a solution $z \in \mathbb{Z}$ if and only if $d \mid b$.

(ii) For every $z \in \mathbb{Z}$, we have $az \equiv 0 \pmod{n}$ if and only if $z \equiv 0 \pmod{n/d}$.

(iii) For all $z, z' \in \mathbb{Z}$, we have $az \equiv az' \pmod{n}$ if and only if $z \equiv z' \pmod{n/d}$.

Proof 3.1: Linear Congruence Identities

Let $a, n \in \mathbb{Z}$ with $n > 0$, and let $d := \gcd(a, n)$.

(i)
$$\begin{aligned} & az \equiv b \pmod{n} \quad \text{for some } z \in \mathbb{Z} \\ \iff\ & az - b = ny \quad \text{for some } z, y \in \mathbb{Z} \quad \text{(Def. of congruence (2.1))} \\ \iff\ & az - ny = b \quad \text{for some } z, y \in \mathbb{Z} \\ \iff\ & d \mid b \quad \text{(By Bezout's Identity (5.2)).} \end{aligned}$$

(ii) Above is Bezout's Identity as a and n form a linear combination of b.-

$$\begin{aligned} n \mid az \iff\ & n/d \mid (a/d)z \text{ (Props. of Divisibility (1.1))} \\ \iff\ & n/d \mid z. \text{ (Cancellation of GCD: } \gcd(a/d, n/d) = 1 \text{ (5.3))} \end{aligned}$$

(iii)
$$\begin{aligned} & az \equiv az' \pmod{n} \\ \iff\ & a(z - z') \equiv 0 \pmod{n} \\ \iff\ & z - z' \equiv 0 \pmod{n/d} \quad \text{(By Part (ii))} \\ \iff\ & z \equiv z' \pmod{n/d}. \end{aligned}$$

■

For emphasis, as we saw above:

Definition 3.1: GCD Reduction

For $a, n \in \mathbb{Z}$, $d := \gcd(a, n)$, then $\gcd(a/d, n/d) = 1$.

Note: "$\to$" (Maps to), "$\mapsto$" (Defines the action of how a single element maps to another), "image" (the set of all outputs), and "pre-images" (the set of all inputs).

A corollary to the above theorem (3.1):

Theorem 3.2: Modular Multiplicative Map

Let $a, n \in \mathbb{Z}$ with $n > 0$, and residue classes $I_n := \{0, \dots, n-1\}$. Then $(a \mod n) \in I_n$. Notably, for $z \in \mathbb{Z}$, $(az \mod n)$ is also in I_n.

I.e., $(az \mod n)$ is some re-ordering of the residue class $(a \mod n)$. Defining function, τ_a:

$$\tau_a : I_n \to I_n : z \mapsto az \bmod n. \tag{3.2.1}$$

The length of the image of τ_a is the number of distinct factors of n relative to a, i.e., n/d. Let the image of τ_a be:

$$E := \{az \mod n : z \in I_n\} = \{i \cdot d \mod n : i = 0, \dots, n/d - 1\}. \tag{3.2.2}$$

The length of the pre-images of τ_a is the number of z solutions to $az \equiv b \pmod n$, i.e., d. Let the pre-images of τ_a be:

$$P := \{z \in I_n : az \equiv b \pmod n\}. \tag{3.2.3}$$

It follows that τ_a is a bijection (one-to-one and onto) if and only if $\gcd(a, n) = 1$. Then, the length of the image is n, and each pre-image has length 1.

Example: for $a = 1, 2, 3, 4, 5, 6$ and $n = 15$,

z	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$2z \mod 15$	0	2	4	6	8	10	12	14	1	3	5	7	9	11	13
$3z \mod 15$	0	3	6	9	12	0	3	6	9	12	0	3	6	9	12
$4z \mod 15$	0	4	8	12	1	5	9	13	2	6	10	14	3	7	11
$5z \mod 15$	0	5	10	0	5	10	0	5	10	0	5	10	0	5	10
$6z \mod 15$	0	6	12	3	9	0	6	12	3	9	0	6	12	3	9

- **Row:2** We see 2 and 15 are coprime, hence n images, $\{0, \dots, n-1\}$.
- **Row:3** We see 3 and 15. Taking out common factors, 15/3, we get 5 distinct images.
- **Row:4** We see 4 and 15 are coprime, hence n images, $\{0, \dots, n-1\}$.
- **Row:5** We see 5 and 15. Taking out common factors, 15/5, we get 3 distinct images.
- **Row:6** We see 6 and 15. Taking out common factors, 15/3, we get 5 distinct images.

Another corollary to the above theorem (3.1):

Theorem 3.3: Modular Congruence Cancellation

Let $a, b, c, n \in \mathbb{Z}$ with $n > 0$ and $\gcd(c, n) = 1$. If $ac \equiv bc \pmod{n}$, then $a \equiv b \pmod{n}$.

Example: We'll demonstrate different representations of members residue class $[2]_5$:

$$\begin{aligned} 8 &\equiv 13 \pmod 5 \quad \text{(i)} \\ 2 \cdot 4 &\equiv 3 \cdot 5 \pmod 5 \quad \text{(ii)} \\ 2 \cdot 4 &\equiv (-3) \cdot 4 \pmod 5 \quad \text{(iii)} \\ 2 &\equiv -3 \pmod 5 \quad \text{(iv)} \end{aligned}$$

Indeed $2 \equiv -3 \pmod 5$, as $2 + 3 \equiv 3 - 3 \pmod 5$. To show this, observe:

a	0	1	2	3	4	5	6	7	8	9	10	11	12
$a \mod 5$	0	1	2	3	4	0	1	2	3	4	0	1	2
	0	-4	-3	-2	-1	0	-4	-3	-2	-1	0	-4	-3

Think of **negative numbers** as traveling backwards within the residue class.

Definition 3.2: Modular Iverses

Let $a, n \in \mathbb{Z}$ with $n > 0$. If $az \equiv 1 \pmod{n}$, then z is the **modular inverse** of a **modulo** n and unique.

Denoted: $a^{-1} \pmod{n}$.

If inverse z modulo n exists, it is unique, as if there were another inverse z', then $z' \equiv z \pmod{n}$.

Restating (3.1) under coprime conditions:

Theorem 3.4: Coprime Modular Multiplicative Identities

Let $a, n \in \mathbb{Z}$ with $n > 0$, and let $\gcd(a, n) = 1$.

(i) The congruence $az \equiv 1 \pmod{n}$ has a solution $z \in \mathbb{Z}$, the modular inverse.

(ii) If $az \equiv 0 \pmod{n}$, then $z \equiv 0 \pmod{n}$ (i.e., z must be a multiple of n).

(iii) If $az \equiv az' \pmod{n}$, then $z \equiv z' \pmod{n}$ (i.e., a cancels out, as long as $\gcd(a, n) = 1$).

Try to find inverses from the above table. Take an a and find solution z to $az \equiv 1 \pmod 5$.

The Chinese Remainder Theorem

Note: $\mathbb{Z}^+$ denotes the set of positive integers, and $\{x_i\}_{i=1}^k$ is short for $\{x_1, ..., x_k\}$.

Theorem 3.5: Chinese Remainder Theorem (CRT)

Let $\{n_i\}_{i=1}^k \in \mathbb{Z}^+$ all be coprime to each other and let $\{a_i\}_{i=1}^k$ be arbitrary integers. Then there is a solution $a \in \mathbb{Z}$ to the system of congruences:

$$\begin{aligned} a &\equiv a_1 \pmod{n_1} \\ a &\equiv a_2 \pmod{n_2} \\ &\vdots \\ a &\equiv a_k \pmod{n_k} \end{aligned}$$

Moreover, if a and b are solutions to the system, then $a \equiv b \pmod{\prod_{i=1}^k n_i}$.

Proof 3.2: Solving a System of Congruences (Part 1)

Let $\{n_i\}_{i=1}^k \in \mathbb{Z}^+$ all be pairwise coprime, and let $\{a_i\}_{i=1}^k$ be arbitrary integers,

Existence: (i) Construct a partial solution for each congruence. (ii) Each partial solution must not interfere with other congruences. (iii) Combine partial solutions:

We define indexes $i, j = 1, ..., k$ representing any two $e_1, ..., e_k$ integers such that:

$$e_j \equiv \begin{cases} 1 \pmod{n_i} & \text{if } j = i, \text{ (target congruence)} \\ 0 \pmod{n_i} & \text{if } j \neq i \text{ (non-interfering).} \end{cases}$$

I.e., e_j has multiplicative identity to it's own system, and additive identity to all other systems by being some multiple. This allows us to construct:

$$\begin{aligned} e_1 \cdot a_1 &\equiv 1 \cdot a_1 \pmod{n_1} \\ e_2 \cdot a_2 &\equiv 1 \cdot a_2 \pmod{n_2} \\ &\vdots \\ e_k \cdot a_k &\equiv 1 \cdot a_k \pmod{n_k} \end{aligned}$$

Using additive identity, we close partial-solutions to $a = \sum_{i=1}^k e_i a_i$, the whole solution. ■

Proof 3.3: Solving a System of Congruences (Part 2)

To construct such $e_1, ..., e_k$, let $n := \prod_{i=1}^{k} n_i$ (the product of all moduli) and $n_i^* := n/n_i$. Then, n_i and n_i^* are coprime, meaning they have solution $n_i^* z \equiv 1 \pmod{n_i}$ for some $z \in \mathbb{Z}$.

Then $z = (n_i^*)^{-1}$, we can now define $e_i := n_i^* z$ for each $i = 1, ..., k$. Therefore, $e_i \equiv 1 \pmod{n_i}$. Since n contains shared factors, and we take n_i at congruence i, $e_i \equiv 0 \pmod{n_j}$ for $i \neq j$.

Thus, we can now construct the solution $a = \sum_{i=1}^{k} e_i a_i$.

■

Proof 3.4: Uniqueness of Solutions (Part 3)

If a and a' both satisfy the system of congruences

$$a \equiv a_i \pmod{n_i} \quad \text{and} \quad a' \equiv a_i \pmod{n_i} \quad \text{for } i = 1, \ldots, k$$

Then they must be congruent, i.e., $a \equiv a' \pmod{\prod_{i=1}^{k} n_i}$.
■

Note: Uniqueness refers to $\mathbb{Z}_n$ (residue classes modulo n), not just a. So there may be multiple solutions, but they congruent to each other under a unique modulus n

Example: We'll find solution a to the system of congruences:

$$\begin{aligned} a &\equiv 3 \pmod{5} \\ a &\equiv 5 \pmod{7} \\ a &\equiv 2 \pmod{11} \end{aligned}$$

Observe that $(3 \bmod 5) = \{3, 8, 13, 18, 23, 28, 33, ...\}$, and $(5 \bmod 7) = \{5, 12, 19, 26, 33, ...\}$. Sets describing $3 + 5k$ and $5 + 7k$ respectively. We see, $3 \equiv 33 \pmod{5}$, and $5 \equiv 33 \pmod{7}$.

Obtaining $e_1 = 7$, as $3 + 5(7) \Longrightarrow 3(7) \equiv 1 \pmod{5}$, and $5 + 7(7) \Longrightarrow 5(7) \equiv 0 \pmod{7}$.
We can take $n = 5 \cdot 7 = 35$ to construct a new system:

$$\begin{aligned} a &\equiv 33 \pmod{35} = 33 + 35k = \{33, 68, ...\} \\ a &\equiv 2 \pmod{11} = 2 + 11k = \{2, 13, 24, 35, 46, 57, 68, ...\} \end{aligned}$$

We see that $33 \equiv 68 \pmod{35}$, and $2 \equiv 68 \pmod{11}$. Thus, $a = 68$:

$$\begin{aligned} 68 &\equiv 3 \pmod{5} \\ 68 &\equiv 5 \pmod{7} \\ 68 &\equiv 2 \pmod{11} \end{aligned}$$

We can design a general algorithm based off this example to solve such systems.

Chinese Remainder Theorem Algorithm

Function 3.1: CRT Algorithm - `crt()`

Computes $a \in \mathbb{Z}$ satisfying a given system of congruences:

Input: Positive integers $\{n_i\}_{i=1}^{k}$ and integers $\{a_i\}_{i=1}^{k}$
Output: An integer a satisfying the system of congruences

```
1  Function crt({n_i}_{i=1}^k, {a_i}_{i=1}^k):
2      a ← a_1;
3      N ← n_1;
4      for i ← 2 to k do
5          while a mod n_i ≠ a_i do
6              a ← a + N;
7          end
8          N ← N × n_i;
9      end
10 return a
```

We compute just like the example above:

1. First take a_1 and n_1 as our initial solution and modulus.
2. Then iterate starting with a_2 and n_2 to find solution $a \bmod (n_i) = a_i$.
3. If a is not congruent to a_i, we increment a by N until it is.
4. Then update N to the new product, and move to the next congruence.

2.4 Residue Classes

We've spoken before about residue classes in (2.4), but we'll go into more detail here.

Theorem 4.1: Residue Intervals

Remainders modular $n \in \mathbb{Z} : n > 1$, denoted $\mathbb{Z}_n$, is the interval $[0, (n-1)]$. As we pass $n-1$, we loop back to 0. Yielding a general interval of $[x, x+(n-1)]$ for $x \in \mathbb{Z}$.

Adding and multiplying residues shifts to some other position in the interval.

- **Addition:** $[(a+b) \bmod n] := [a] + [b] = [a+b] = [c] \iff a + b \equiv c \pmod{n}$
- **Multiplication:** $[(a \cdot b) \bmod n] := [a] \cdot [b] = [a \cdot b] = [c] \iff a \cdot b \equiv c \pmod{n}$

If n is odd, then our interval is $[-(n-1)/2, (n-1)/2]$. If even, then $[-n/2, n/2-1]$.

Example: Consider tables $\mathbb{Z}_5$ and $\mathbb{Z}_6$:

a	0	1	2	3	4	5	6	7	8	9	10	11	12
$a \mod 5$	0	1	2	3	4	0	1	2	3	4	0	1	2
	0	-4	-3	-2	-1	0	-4	-3	-2	-1	0	-4	-3

Since 5 is odd, our interval is $[-(4)/2, (4)/2] = [-2, 2]$, which could be seen as the interval $a \in [3, 7]$.

a	0	1	2	3	4	5	6	7	8	9	10	11	12
$a \mod 6$	0	1	2	3	4	5	0	1	2	3	4	5	0
	0	-5	-4	-3	-2	-1	0	-5	-4	-3	-2	-1	0

Since 6 is even, our interval is $[-6/2, 6/2-1] = [-3, 2]$, which could be seen as the interval $a \in [3, 8]$. This interval is no different than $[0, 5]$ or $[0, 6]$, this shifting of the interval captures $[x, x + (n-1)]$.

Note: We'll use α: "alpha"; β: "beta"; and such as variables when discussing residue classes.

Theorem 4.2: Residue Class Operations

Let $\alpha \in \mathbb{Z}_n$ be residue classes. Then:

- **Additive Identity:** $\alpha + [0] = \alpha$; **Additive Inverse:** $\alpha + (-\alpha) = [0]$.
- **Multiplicative Identity:** $\alpha \cdot [1] = \alpha$; **Multiplicative Inverse:** $\alpha \cdot \alpha^{-1} = [1]$.

Moreover, **Residue classes form a ring** (3.3), including distributive properties.

Theorem 4.3: Inverse Residue Classes

For $n \in \mathbb{Z} : n > 1$,
let $Z_n^* := \{\alpha \in \mathbb{Z}_n \mid \gcd(\alpha, n) = 1\}$, i.e., Z_n^* contains elements in $\mathbb{Z}_n$ where α^{-1} exists.

- If n is prime, then $Z_n^* = \mathbb{Z}_n \setminus \{[0]\}$, i.e., Z_n^* contains all elements in $\mathbb{Z}_n$ except $[0]$.
- If n is composite, then $Z_n^* \subsetneq \mathbb{Z}_n \setminus \{[0]\}$.

Moreover, Z_n^* forms a group under multiplication. Therefore for all $\beta \in Z_n^*$, we have $\alpha\beta \in Z_n^*$.

Note: The symbol $\subsetneq$ denotes a proper subset. If $A \subsetneq B$, then A is a subset of B but not equal to B.

Proof 4.1: Residue Class Inverses

Primes: The congruence $\alpha z \equiv 1 \pmod{n}$ has a solution z for all $\alpha \in \mathbb{Z}_n$ if $gcd(\alpha, n) = 1$ (3.4).

Composites: $Z_n^* \subsetneq \mathbb{Z}_n \setminus \{[0]\}$. If $d := gcd(\alpha, n) \mid n$, and $1 < d < n$, then $d \neq 0$ and $\alpha \notin Z_n^*$ (3.1). We say $d < n$, otherwise $n \equiv 0 \pmod{n}$ where $d = n$.

Multiplicative Group:

- **Inverse:** Every element in Z_n^* has an inverse.
- **Closure:** Therefore $\alpha\beta \in Z_n^*$, as $(\alpha^{-1})\alpha\beta \equiv \beta \pmod{n}$ and $(\beta^{-1})\beta\alpha \equiv \alpha \pmod{n}$.

■

Theorem 4.4: Inverse Operations

Let $\alpha, \beta, \gamma \in \mathbb{Z}_n$ be residue classes. Then:

- **Inverse of Inverse:** $(\alpha^{-1})^{-1} = \alpha$
- **Product of Inverse:** $(\alpha \cdot \beta)^{-1} = \alpha^{-1} \cdot \beta^{-1}$
- **Inverse Division:** $\alpha/\beta = \alpha \cdot \beta^{-1}$
- **Cancellation Law:** $\alpha\beta = \alpha\gamma \implies \beta = \gamma \iff \alpha \in Z_n^*$.

Theorem 4.5: Residue Powers Identities

Powers work similarly to integers. For $\alpha, \beta \in \mathbb{Z}_n$ and $k, l \in \mathbb{Z}$, which also hold for $\alpha, \beta \in \mathbb{Z}_n^*$:

- **Zero Power:** $\alpha^0 = [1]$
- **General Powers:** $\alpha^1 = \alpha$ and $\alpha^2 = \alpha \cdot \alpha$ and so on.
- **Inverse Power:** Inverse α^k is $(\alpha^{-1})^k$.
- **Power of a Power:** $(\alpha^l)^k = \alpha^{lk} = (\alpha^k)^l$.
- **Product of Powers:** $\alpha^k \cdot \alpha^l = \alpha^{k+l}$.
- **Quotient of Powers:** $\alpha^k/\alpha^l = \alpha^{k-l}$.
- **Power of a Product:** $(\alpha\beta)^k = \alpha^k \cdot \beta^k$.

We may now generalize the Chinese Remainder Theorem (3.5) under residue classes.

Theorem 4.6: Chinese Remainder Map

Let $\{n_i\}_{i=1}^k \in \mathbb{Z}^+$ all be pairwise coprime, and $n := \prod_{i=1}^k n_i$. We define the map:

$$\begin{aligned} \theta : \mathbb{Z}_n &\to \mathbb{Z}_{n_1} \times \cdots \times \mathbb{Z}_{n_k} \\ [a]_n &\mapsto ([a]_{n_1}, \ldots, [a]_{n_k}) \end{aligned}$$

For $\mathbb{Z}_n$ (Residue classes modulo n), we can visualize:

$$\theta([a]_n) = \begin{cases} [a]_{n_1} & \mod n_1 \\ [a]_{n_2} & \mod n_2 \\ \vdots & \vdots \\ [a]_{n_k} & \mod n_k \end{cases}$$

Where $[a]_n$ can be thought of as our a solution in the system of congruences:

$$\begin{aligned} a &\equiv a_1 \pmod{n_1} \\ a &\equiv a_2 \pmod{n_2} \\ &\vdots \\ a &\equiv a_k \pmod{n_k}, \end{aligned}$$

extending the Chinese Remainder Theorem to classes produced by a mod n, not just a.

(i) θ is unambiguous, i.e., any $[a]_n \in \mathbb{Z}_n$ has a unique image in $\mathbb{Z}_{n_1} \times \cdots \times \mathbb{Z}_{n_k}$.

(ii) θ forms a ring isomorphism, meaning:

- (a) θ is a bijection (one-to-one and onto), i.e., there's an inverse map θ^{-1}, which is the process of finding a from $[a]_n$ (The Chinese Remainder Theorem).
- (b) θ preserves addition and multiplication, since residues form a ring. Thus, operating on residue classes only affects the inputs to the map (4.2).

Tip: The Chinese Remainder Map (θ) generates a system of congruences, while the Chinese Remainder Theorem solves them (θ^{-1}).

Euler's Phi Function

Tip: Leonhard Euler (1707–1783), pronounced as "oiler," was a Swiss mathematician born in Basel. He worked in St. Petersburg and Berlin, shaping calculus and number theory.

Also known as the **Euler Totient Function**:

Definition 4.1: Euler's Phi Function

For all $n \in \mathbb{Z}^+$, we define Euler's Phi Function as:

$$\varphi(n) := |\mathbb{Z}_n^*|$$

The number of inverses modulo n. Numbers coprime to n are in $\mathbb{Z}_n^*$.
Therefore, for primes p, $\varphi(p) = p - 1$.

Theorem 4.7: Chinese Remainder's Phi Function

Let $n := \prod_{i=1}^{k} n_i$ be the product of pairwise coprime integers. Then:

$$\varphi(n) = \prod_{i=1}^{k} \varphi(n_i) = \varphi(n_1) \cdot \varphi(n_2) \cdots \varphi(n_k)$$

The number of inverses in $\mathbb{Z}_n^*$ is the product of the number of inverses in $\mathbb{Z}_{n_i}^*$.

Proof 4.2: Chinese Remainder's Phi Function

Consider the Chinese Remainder Map $\theta : \mathbb{Z}_n \to \mathbb{Z}_{n_1} \times \mathbb{Z}_{n_2} \times \cdots \times \mathbb{Z}_{n_k}$. Since θ is isomorphic, it has a one-to-one correspondence. If we restrict our input to $\mathbb{Z}_n^*$, then the output will be in $\mathbb{Z}_{n_1}^* \times \mathbb{Z}_{n_2}^* \times \cdots \times \mathbb{Z}_{n_k}^*$. Hence, $|\mathbb{Z}_n^*| = |\mathbb{Z}_{n_1}^*| \times |\mathbb{Z}_{n_2}^*| \times \cdots \times |\mathbb{Z}_{n_k}^*| = \prod_{i=1}^{k} |\mathbb{Z}_{n_i}^*| = \prod_{i=1}^{k} \varphi(n_i)$. ■

Theorem 4.8: Euler's Phi of a Raised Prime

Let p be a prime and $e \in \mathbb{Z}^+$. Then:

$$\varphi(p^e) = p^{e-1}(p-1)$$

Proof 4.3: Euler's Phi of a Raised Prime

$\varphi(n)$ counts residue classes in $\mathbb{Z}_n$ that are coprime to n. $\mathbb{Z}_n$ represent integers $[0, n-1]$.

Examining $\mathbb{Z}_{p^e}$, to obtain coprimes, we omit members sharing common factors to p^e, i.e., multiples p, which p^e gives us e of.

Since the last factor reaches p^e, we ignore it, as it's beyond $p^e - 1$. Leaving us p^{e-1} multiples. Therefore, $\varphi(p^e) = p^e - p^{e-1} = p^{e-1}(p-1)$.

■

As implied by Theorem 4.7, we can generalize this to the prime factorization of n.

Theorem 4.9: Phi of Prime Factorization

Let $n := \prod_{i=1}^{k} p_i^{e_i}$ be the prime factorization of n. $\{p_i^{e_i}\}$ are pairwise coprime. Then:

$$\varphi(n) = \prod_{i=1}^{k} p_i^{e_i - 1}(p_i - 1)$$

Expanding the product,

$$\varphi(n) = p_1^{e_1} \cdot \left(1 - \frac{1}{p_1}\right) \cdot p_2^{e_2} \cdot \left(1 - \frac{1}{p_2}\right) \cdot \cdots \cdot p_k^{e_k} \cdot \left(1 - \frac{1}{p_k}\right)$$

Which gives us:

$$\varphi(n) = n \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right)$$

as n represents $p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k}$.

2.5 Euler's Theorem & Fermat's Little Theorem

We know residues repeat in $\mathbb{Z}_n$ after n steps, forming a cycle. We've been used to seeing such cycles end and start at 0. However, when we restrict ourselves to $\mathbb{Z}_n^*$, 0 is excluded. We'll find that cycles in $\mathbb{Z}_n^*$ jump by powers of $\alpha \in \mathbb{Z}_n^*$, starting and ending at 1.

Definition 5.1: Multiplicative Order

Let $n \in \mathbb{Z}^+$ and $a \in \mathbb{Z}_n^*$. The multiplicative order of a modulo n is the smallest positive integer k such that $a^k \equiv 1 \pmod{n}$.

Theorem 5.1: Multiplicative Order Interval

Let $n \in \mathbb{Z}^+$ and $\alpha \in \mathbb{Z}_n^*$. The multiplicative order k repeats every k steps. Therefore, for every index:

- $i \in \mathbb{Z}$, $\alpha^i \equiv 1 \pmod{n} \iff k \mid i$, i.e., $i \equiv 0 \pmod{k}$.
- $i, j \in \mathbb{Z}$, $\alpha^i \equiv \alpha^j \pmod{n} \iff i \equiv j \pmod{k}$.

Example: Let $n = 7$ and take $\alpha = 1, \ldots, 6$.

- $\alpha = 1$: order 1.
- $\alpha = 2$: order 3.
- $\alpha = 3$: order 6.
- $\alpha = 4$: order 3.
- $\alpha = 5$: order 6.
- $\alpha = 6$: order 2.

i	1	2	3	4	5	6
$1^i \bmod 7$	1	1	1	1	1	1
$2^i \bmod 7$	2	4	1	2	4	1
$3^i \bmod 7$	3	2	6	4	5	1
$4^i \bmod 7$	4	2	1	4	2	1
$5^i \bmod 7$	5	4	6	2	3	1
$6^i \bmod 7$	6	1	6	1	6	1

We see that $\alpha = 2$ for $i = 3$ and $i = 6$, 3 is the smallest k such that $2^k \equiv 1 \pmod{7}$. Additionally, we see the relationship $2^i \equiv 2^j \pmod{7}$ if and only if $i \equiv j \pmod{3}$.

Note: For set S, $\prod_{\beta \in S} \beta$ is the product of all elements in S.

Theorem 5.2: Euler's Theorem

Let $n \in \mathbb{Z}^+$ and $\alpha \in \mathbb{Z}_n^*$. Then $\alpha^{\varphi(n)} \equiv 1 \pmod{n}$, when multiplicative order α divides $\varphi(n)$.

Proof 5.1: Euler's Theorem

For every $\beta \in \mathbb{Z}_n^*$, theres an $\alpha \in \mathbb{Z}_n^*$ such that $\alpha\beta \in \mathbb{Z}_n^*$ (4.2) $\varphi(n)$ and$\mathbb{Z}_n^*$:

$$\prod_{\beta \in \mathbb{Z}_n^*} \beta = \prod_{\beta \in \mathbb{Z}_n^*} \alpha\beta = \alpha^{\varphi(n)} \prod_{\beta \in \mathbb{Z}_n^*} \beta$$

Taking the inverse of $\prod_{\beta \in \mathbb{Z}_n^*} \beta$ results in $1 = \alpha^{\varphi(n)}$. **Note:** $\varphi(n) := |\mathbb{Z}_n^*|$ and both products β and $\alpha\beta$ produce the same set as we cycle inverse residues.

■

Theorem 5.3: Fermat's Little Theorem (FLT)

For every prime p and residue classes $\alpha \in \mathbb{Z}_p^*$: $\alpha^p \equiv \alpha \pmod{p}$.

Proof 5.2: Fermat's Little Theorem

Since p is prime, $\varphi(p) = p-1$. By Euler's Theorem, $\alpha^{p-1} \equiv 1 \pmod{p}$. Therefore, multiplying α to both sides yields, $\alpha^{p-1} \cdot \alpha \equiv 1 \cdot \alpha \pmod{p}$. Hence $\alpha^p \equiv \alpha \pmod{p}$. ■

Definition 5.2: Primitive Root

Let $n \in \mathbb{Z}^+$ and $\alpha \in \mathbb{Z}_n^*$. If the multiplicative order of α modulo n is $\varphi(n)$, then α is a primitive root modulo n.

Theorem 5.4: Multiplicative Order of Powers

If $\alpha \in \mathbb{Z}_n^*$ has multiplicative order k. Then from every new residue produced by α^m where $m \in \mathbb{Z}$, the multiplicative order of α^m is:

$$\frac{k}{\gcd(m,k)}$$

Example: Let $n = 7$ and $\alpha = 1, \ldots, 6$.

- $\alpha = 2^1 = 2$: has order $\dfrac{3}{\gcd(1,3)} = 3$.
- $\alpha = 2^2 = 4$: has order $\dfrac{3}{\gcd(2,3)} = 3$.
- $\alpha = 2^3 = 8 = 1$: has order $\dfrac{3}{\gcd(3,3)} = 1$.

i	1	2	3	4	5	6
$1^i \bmod 7$	1	1	1	1	1	1
$2^i \bmod 7$	2	4	1	2	4	1
$3^i \bmod 7$	3	2	6	4	5	1
$4^i \bmod 7$	4	2	1	4	2	1
$5^i \bmod 7$	5	4	6	2	3	1
$6^i \bmod 7$	6	1	6	1	6	1

Raising $\alpha = 2^3$ gave us 8, which is congruent to 1 modulo 7, and $\alpha = 1$ has order 1. Moreover, residues 3 and 5 are primitive roots.

We will abstract variables to emphasize α^m being some other residue after shifting by m.

Proof 5.3: Multiplicative Order of Powers

We define $\alpha^m := \beta$. Then β's multiplicative order is the smallest l such that $\beta^l \equiv 1 \pmod{n}$. Then by (5.1),

$$\alpha^{m \cdot l} \equiv 1 \pmod{n} \iff ml \equiv 0 \pmod{k}$$

We can drop m as a common factor by taking $\gcd(m, k)$ from k (3.1), yielding:

$$l \equiv 0 \pmod{\frac{k}{\gcd(m,k)}}$$

Thus we have $\beta^l \equiv 1 \pmod{n} \iff l \equiv 0 \pmod{\frac{k}{\gcd(m,k)}}$ satisfying the definition.

■

2.6 Quadratic Residues

Quadratic residues pertain to congruences of form $z^2 \equiv a \pmod{p}$, where p is prime. Though we start with general observations of residues produced by powers.

Definition 6.1: Residue Classes of Powers

We shall extend $\mathbb{Z}_n^*$ to powers, such that for all $\beta \in \mathbb{Z}_n^*$:

$$(\mathbb{Z}_n^*)^m := \{\beta^m \pmod{n}\}$$

The set $(\mathbb{Z}_n^*)^m$ from $\mathbb{Z}_n^*$ at the very least holds $[1]_n$.

To illustrate our definition, we'll re-use our previous example $(\mathbb{Z}_7^*)^m$:

m	$(\mathbb{Z}_7^*)^1$	$(\mathbb{Z}_7^*)^2$	$(\mathbb{Z}_7^*)^3$	$(\mathbb{Z}_7^*)^4$	$(\mathbb{Z}_7^*)^5$	$(\mathbb{Z}_7^*)^6$
$1^m \bmod 7$	1	1	1	1	1	1
$2^m \bmod 7$	2	4	1	2	4	1
$3^m \bmod 7$	3	2	6	4	5	1
$4^m \bmod 7$	4	2	1	4	2	1
$5^m \bmod 7$	5	4	6	2	3	1
$6^m \bmod 7$	6	1	6	1	6	1

We see $(\mathbb{Z}_n^*)^1 = \{1, 2, 3, 4, 5, 6\}$, $(\mathbb{Z}_n^*)^2 = \{1, 2, 4\}$, $(\mathbb{Z}_n^*)^3 = \{1, 6\}$, annd so on.

For emphasis of our definition:

Theorem 6.1: Intersection of $\mathbb{Z}_n^*$ Powers

Let, $\alpha, \beta \in \mathbb{Z}_n^*$ and all $l, m \in \mathbb{Z}$,
If $\alpha^l \equiv \beta^m \pmod{n}$ then their residue r is in both $(\mathbb{Z}_n^*)^m$ and in $(\mathbb{Z}_n^*)^l$.

Example: As $(\mathbb{Z}_n^*)^1 = \{1, 2, 3, 4, 5, 6\}$, $(\mathbb{Z}_n^*)^2 = \{1, 2, 4\}$, both have 1,2, and 4 in common.

Theorem 6.2: Properties of Powers $(\mathbb{Z}_n^*)^m$

Let n be a positive integer, let $\alpha, \beta \in \mathbb{Z}_n^*$, and let m be any integer.

(i) If $\alpha \in (\mathbb{Z}_n^*)^m$, then $\alpha^{-1} \in (\mathbb{Z}_n^*)^m$.

(ii) If $\alpha \in (\mathbb{Z}_n^*)^m$ and $\beta \in (\mathbb{Z}_n^*)^m$, then $\alpha\beta \in (\mathbb{Z}_n^*)^m$.

(iii) If $\alpha \in (\mathbb{Z}_n^*)^m$ and $\beta \notin (\mathbb{Z}_n^*)^m$, then $\alpha\beta \notin (\mathbb{Z}_n^*)^m$.

Proof 6.1: Properties of Powers $(\mathbb{Z}_n^*)^m$

Let n be a positive integer, let $\alpha, \beta \in \mathbb{Z}_n^*$, and let m be any integer.

(i) If $\alpha \equiv \gamma^m \pmod{n}$, then $\alpha^{-1} \equiv (\gamma^{-1})^m \pmod{n}$

(ii) If $\alpha \equiv \gamma^m \pmod{n}$ and $\beta \equiv \delta^m \pmod{n}$, then $\alpha\beta \equiv (\gamma\delta)^m \pmod{n}$

(iii) Assume $\alpha \in (\mathbb{Z}_n^*)^m$, $\beta \notin (\mathbb{Z}_n^*)^m$, and $\alpha\beta \in (\mathbb{Z}_n^*)^m$. Then by (i), $\alpha^{-1} \in (\mathbb{Z}_n^*)^m$, we have by (ii), $\beta = \alpha^{-1}(\alpha\beta) \equiv \beta \pmod{n}$; However $\beta \notin (\mathbb{Z}_n^*)^m$, which is a contradiction.

■

Note: For $\alpha := [a] \in \mathbb{Z}_n$ and $b \in \mathbb{Z}$, we will often switch between $\alpha = b$ and $\alpha \equiv b \pmod{n}$, as α represents an element for which either equality or congruence holds.

Tip: Pierre de Fermat (1601-1665) was a French lawyer and mathematician. Born in Beaumont-de-Lomagne, France, Fermat is best known for his work in number theory, analytic geometry, and probability. His famous "Fermat's Last Theorem" remained unsolved for over 350 years. He claimed $an + bn = cn : n > 2$ has no integer solution.

Theorem 6.3: Coprime Powers $\mathbb{Z}_n^*$

Let $n \in \mathbb{Z}^+$, $\alpha \in \mathbb{Z}_n^*$ and all $l, m \in \mathbb{Z}$. Then:

$$\gcd(l, m) = 1 \text{ and } \alpha^l \in (\mathbb{Z}_n^*)^m \implies \alpha \in (\mathbb{Z}_n^*)^m$$

Proof 6.2: Coprime Powers $\mathbb{Z}_n^*$

For $\alpha^l \in (\mathbb{Z}_n^*)^m$ to exists means, $\alpha^l = \beta^m \pmod n$ for some $\beta \in \mathbb{Z}_n^*$. Since $\gcd(l, m) = 1$, by Bezout's Identity, there exists $ls + mt = 1$ for some $s, t \in \mathbb{Z}$.

$$a^1 = a^{ls+mt} = a^{ls}a^{mt} = \left(a^l\right)^s a^{mt} = (\beta^m)^s \alpha^{mt} = \left(\beta^s \alpha^t\right)^m =: \gamma^m$$

Therefore, $\alpha = \gamma^m \pmod n$, thus $\alpha \in (\mathbb{Z}_n^*)^m$. ■

Definition 6.2: Quadratic Residue

Let $\alpha, n \in \mathbb{Z}$,$\beta \in \mathbb{Z}_n^*$, and $n > 1$. Then,

- "α **is a quadratic residue modulo n**" if $\gcd(\alpha, n) = 1$ and $\alpha \equiv \beta^2 \pmod n$.
- "α **is a quadratic non-residue modulo n**" if $\gcd(\alpha, n) = 1$ and $\alpha \not\equiv \beta^2 \pmod n$.
- "β **is a square root of** α **modulo n**" if $\beta^2 \equiv \alpha \pmod n$.

Quadratic Residues Modulo Primes

We shall consider odd primes p, i.e., primes greater than 2, as 2 creates special cases that can complicate some of our results on quadratic residues.

Theorem 6.4: Square Roots 1 Modulo p

Let p be an odd prime, and $\beta \in \mathbb{Z}_p$. Then $\beta^2 = 1 \iff \beta = \pm 1$.

Proof 6.3: Square Roots of 1 Modulo p

For $\beta \in \mathbb{Z}_p$, if $\beta = \pm 1$ then $\pm 1^2 \equiv 1 \pmod p$. If $\beta^2 = 1$, then $\beta^2 - 1 \equiv 0 \pmod p$, and $p \mid (b^2 - 1)$. For difference of squares, $(b^2 - 1) = (b + 1)(b - 1)$, and since p is prime, $p \mid (b + 1)$ or $p \mid (b - 1)$. Thus, $\beta \equiv \pm 1 \pmod p$. ■

To reduce repetition, we will use $(\mathbb{Z}_p^*)^2$, to denote Quadratic Residues modulo odd primes p.

Theorem 6.5: Square Roots $(\mathbb{Z}_p^*)^2$

Let p be an odd prime, for $\gamma, \beta \in \mathbb{Z}_p^*$. Then $\gamma^2 \equiv \beta^2 \pmod{p} \iff \gamma \equiv \pm\beta \pmod{p}$.
I.e., for some $\alpha \in \mathbb{Z}_p^*$, if $\alpha = \beta^2$, then α has two square roots modulo p: $\pm\beta$.

Proof 6.4: Square Roots $(\mathbb{Z}_p^*)^2$

Following from the theorem proof we have,

$$\gamma^2 = \beta^2 \iff \frac{\gamma^2}{\beta^2} = 1 \iff \frac{\gamma}{\beta} = \pm 1 \iff \gamma = \pm\beta$$

In terms of congruences, members of $\mathbb{Z}_p^*$, γ and β are invertible:

$$\begin{aligned} \gamma^2 \equiv \beta^2 \pmod{p} &\iff \gamma^2 \cdot \beta^{-2} \equiv 1 \pmod{p} \\ &\iff \gamma \cdot \beta^{-1} \equiv \pm 1 \pmod{p} \\ &\iff \gamma \equiv \pm\beta \pmod{p} \end{aligned}$$

■

Theorem 6.6: Cardinality of $(\mathbb{Z}_p^*)^2$

Let p be an odd prime. Then,

$$|(\mathbb{Z}_p^*)^2| = \frac{(p-1)}{2}$$

Allowing us to represent $(\mathbb{Z}_p^*)^2$ as the interval $[1, (p-1)/2]$.

Proof 6.5: Cardinality of $(\mathbb{Z}_p^*)^2$

Let $\beta^2 \in (\mathbb{Z}_p^*)^2$, then there are two square roots for β^2: $\pm\beta$. We define the map:

$$\begin{aligned} \sigma : \mathbb{Z}_p^* &\to \mathbb{Z}_p^* \\ \pm\beta &\mapsto \beta^2 \bmod p \end{aligned}$$

Since σ is two-to-one where two elements from $\mathbb{Z}_p^*$ map to one element $(\mathbb{Z}_p^*)^2$, we cut our output in half. Hence, $|(\mathbb{Z}_p^*)^2| = \frac{|\mathbb{Z}_p^*|}{2} = \frac{p-1}{2}$. ■

Theorem 6.7: Euler's Criterion (Taking Square Root $(\mathbb{Z}_p^*)^2$)

Let p be an odd prime and $\alpha \in \mathbb{Z}_p^*$. Then,

(i) $\alpha^{(p-1)/2} \equiv \pm 1 \pmod{p}$

(ii) If $\alpha \in (\mathbb{Z}_p^*)^2$, then $\alpha^{(p-1)/2} \equiv 1 \pmod{p}$.

(iii) If $\alpha \notin (\mathbb{Z}_p^*)^2$, then $\alpha^{(p-1)/2} \equiv -1 \pmod{p}$.

Note: Remember for $a \in \mathbb{Z}$, that $a^{\frac{1}{2}}$ is cancelled by $(a^{\frac{1}{2}})^2 = a$.

Proof 6.6: Euler's Criterion

(i) Let $\gamma = \alpha^{(p-1)/2}$. Then $\gamma^2 = \alpha^{p-1} = 1$ (5.3). Then $\gamma = \pm 1$ (6.4).

(ii) If $\alpha \in (\mathbb{Z}_p^*)^2$, $\alpha = \beta^2 \in \mathbb{Z}_p^*$. Then $(\beta^2)^{(p-1)/2} = \alpha^{(p-1)/2}$. Then , $\beta^{p-1} \equiv 1 \pmod{p}$ (5.3).

(iii) Examine $|(\mathbb{Z}_p^*)^2| = (p-1)/2$. Then the other half of $\mathbb{Z}_p^*$ are quadratic non-residues, for which we define the set $\mathcal{P}$. We describe $\mathcal{P}$'s image by map μ:

$$\begin{aligned} \mu : \mathbb{Z}_p^* &\to \mathbb{Z}_p^* \\ \kappa\lambda &\mapsto \alpha \notin (\mathbb{Z}_p^*)^2 \end{aligned}$$

For $\kappa, \lambda \in \mathbb{Z}_p^* : \kappa \neq \lambda$. Then, $\lambda = \frac{\alpha}{\kappa}$, so λ is uniquely determined by κ. We define the set $C := \{\kappa\lambda \in \mathbb{Z}_p^* : \kappa\lambda = \alpha\}$, and $C \subseteq \mathcal{P}$. Then their product

$$\prod_{\{\kappa\lambda \in C\}} \kappa\lambda \;=\; \prod_{\{\kappa\lambda \in C\}} \alpha = \alpha^{(p-1)/2}$$

as a result of $\mathcal{P}$ containing $(p-1)/2$ many $\kappa\lambda$ pairs. We define $\epsilon := \prod_{\{\kappa\lambda \in C\}} \kappa\lambda$, and partition $D := \{\{\kappa\lambda\} \in \mathcal{P} : \kappa\lambda = 1\}$. If $\kappa\lambda = 1$, then $\kappa := \lambda^{-1}$, and κ uniquely determines λ. then $\kappa = \lambda$ if and only if $\kappa^2 = 1$, which implies $\kappa = \pm 1$ (6.4). So we exclude $[\pm 1]$ from our set D. Still for other pairs, $\kappa\lambda = 1$, we proceed:

$$\epsilon = \; [1] \cdot [-1] \prod_{\{\kappa\lambda \in D\}} \kappa\lambda \;=\; [-1] \prod_{\{\kappa\lambda \in D\}} [1] = -1$$

As we bring back $[\pm 1]$ in the product, we see $\epsilon = -1$. Thus $\alpha^{(p-1)/2} \equiv -1 \pmod{p}$.

■

To abstract our findings we define the Legendre Symbol.

Function 6.1: Legendre Symbol

Let p be an odd prime, and $\alpha \in \mathbb{Z}_p^*$. Then the Legendre Symbol is defined as:

$$\left(\frac{\alpha}{p}\right) = \begin{cases} 1 & \text{if } \alpha \in (\mathbb{Z}_p^*)^2 \ (\alpha \text{ is a quadratic residue}), \\ -1 & \text{if } \alpha \notin (\mathbb{Z}_p^*)^2 \ (\alpha \text{ is a quadratic non-residue}). \\ 0 & \text{if } \alpha \equiv 0 \pmod{p} \ (p \mid \alpha \text{ and has no inverse}). \end{cases}$$

Shortening our notation from (ii) and (iii) in Euler's Criterion (6.7).

This next theorem is a direct result from part (iii) in our proof of Euler's Criterion (6.7).

Theorem 6.8: Wilson's Theorem

Let p be an odd prime, then

$$(p-1)! \equiv -1 \pmod{p}$$

Proof 6.7: Wilson's Theorem

Let p be an odd prime. We know each element $\kappa, \lambda \in \mathbb{Z}_p^* : \kappa \neq \lambda$, has an inverse $\kappa\lambda = 1$. Except for $[\pm 1]$, as 1 is its own inverse.

We take $(p-1)! = (p-1)(p-2)\cdots(2)(1)$, and pair each element with its inverse:

$$(p-1)! \equiv (2 \cdot 2^{-1})(3 \cdot 3^{-1}) \cdots (1 \cdot (p-1)) \pmod{p}$$

Note, $(p-1) = -1$, as -1 congruently is equivalent to the last element, i.e., $p-1$, yielding:

$$(p-1)! \equiv (1)(1) \cdots (1 \cdot (p-1)) \pmod{p}$$

Thus, $(p-1)! \equiv (p-1) \pmod{p}$, which is $(p-1)! \equiv -1 \pmod{p}$.

■

Theorem 6.9: Quadratic Non-Residues Product $\mathbb{Z}_p^*$

Let p be an odd prime and $\alpha, \beta \in \mathbb{Z}_p^*$. If $\alpha \notin (\mathbb{Z}_p^*)^2$ and $\beta \notin (\mathbb{Z}_p^*)^2$, then $\alpha\beta \in (\mathbb{Z}_p^*)^2$.

Proof 6.8: Quadratic Non-Residues' Product $\mathbb{Z}_p^*$

For p an odd prime, let $\alpha, \beta \in \mathbb{Z}_p^* : \alpha, \beta \notin (\mathbb{Z}_p^*)^2$. Then,

$$\left(\frac{\alpha}{p}\right) = -1 \text{ and } \left(\frac{\beta}{p}\right) = -1$$

Therefore,

$$(\alpha\beta)^{(p-1)/2} = \left(\frac{\alpha}{p}\right) \cdot \left(\frac{\beta}{p}\right) = [-1] \cdot [-1] = 1$$

Thus $\alpha\beta \in (\mathbb{Z}_p^*)^2$ by Euler's Criterion (6.7). ■

Giving us the following theorem:

Theorem 6.10: Products Quadratic Residues and Non-Residues

For odd p primes and $\alpha \in (\mathbb{Z}_p^*)^2$, $\beta \notin (\mathbb{Z}_p^*)^2$, then:

$$\left(\frac{\alpha}{p}\right) \times \left(\frac{\alpha}{p}\right) \in (\mathbb{Z}_p^*)^2$$
$$\left(\frac{\alpha}{p}\right) \times \left(\frac{\beta}{p}\right) \notin (\mathbb{Z}_p^*)^2$$
$$\left(\frac{\beta}{p}\right) \times \left(\frac{\beta}{p}\right) \in (\mathbb{Z}_p^*)^2$$

Which is to say, $[1] \cdot [1] = 1$, $[1] \cdot [-1] = -1$, and $[-1] \cdot [-1] = 1$.

Quadratic Residues Modulo of Raised Primes

Theorem 6.11: Square Roots 1 Modulo p^e

Let p be an odd prime, $e \in \mathbb{Z}^+$, and $\beta \in \mathbb{Z}_{p^e}$. Then $\beta^2 = 1 \iff \beta = \pm 1$.

Proof 6.9: Square Roots 1 Modulo p^e

If $\beta = \pm 1$, then $\pm 1^2 \equiv 1 \pmod{p^e}$. If $\beta^2 = 1$, then $\beta^2 - 1 \equiv 0 \pmod{p^e}$, and $p^e \mid (\beta^2 - 1)$. Then p divides $(\beta + 1)$ or $(\beta - 1)$. Note, if p divides both, p divides their difference, 2. This is impossible as p is odd. Since, p divides either $(\beta + 1)$ or $(\beta - 1)$, then $\beta = \pm 1$. ■

It follows that our previous theorems $\mathbb{Z}_p^*$ follow to $\mathbb{Z}_{p^e}^*$. We shall continue without proof.

Theorem 6.12: Square Roots $\mathbb{Z}_{p^e}^*$

Let p be an odd prime and $e \in \mathbb{Z}^+$, for $\gamma, \beta \in \mathbb{Z}_{p^e}^*$. Then $\gamma^2 = \beta^2 \iff \gamma = \pm\beta$.

Theorem 6.13: Cardinality of $\mathbb{Z}_{p^e}^*$

Let p be an odd prime. Then, $|\mathbb{Z}_{p^e}^*| = \frac{\varphi(p^e)}{2}$.

Theorem 6.14: Euler's Criterion (Taking Square Root $\mathbb{Z}_{p^e}^*$)

Let p be an odd prime and $\alpha \in \mathbb{Z}_{p^e}^*$. Then,

(i) $\alpha^{\varphi(p^e)/2} \equiv \pm 1 \pmod{p}$

(ii) If $\alpha \in \mathbb{Z}_{p^e}^*$, then $\alpha^{\varphi(p^e)/2} \equiv 1 \pmod{p}$.

(iii) If $\alpha \notin \mathbb{Z}_{p^e}^*$, then $\alpha^{\varphi(p^e)/2} \equiv -1 \pmod{p}$.

Note: $\varphi(p^e) = p^e - p^{e-1} = p^{e-1}(p-1)$.

Theorem 6.15: Quadratic Non-Residues Product $\mathbb{Z}_{p^e}^*$

Let p be an odd prime and $\alpha, \beta \in \mathbb{Z}_{p^e}^*$. If $\alpha \notin (\mathbb{Z}_n^*)^2$ and $\beta \notin (\mathbb{Z}_n^*)^2$, then $\alpha\beta \in (\mathbb{Z}_n^*)^2$.

Leading us to the following theorem:

Theorem 6.16: Quadratic Congugrences $\mathbb{Z}_{p^e}^* \to (\mathbb{Z}_p^*)^2$

Let p be an odd prime, $e \in \mathbb{Z}^+$, and $\alpha \in \mathbb{Z}$. Then,

$$\left(\frac{\alpha}{p^e}\right) = \pm 1 \iff \left(\frac{\alpha}{p}\right) = \pm 1$$

Proof 6.10: Quadratic Congugrences $\mathbb{Z}_{p^e}^* \to (\mathbb{Z}_p^*)^2$

Let p be an odd prime, $e \in \mathbb{Z}^+$, and $\alpha \in \mathbb{Z}$,

Proving $\left(\frac{\alpha}{p^e}\right) = \pm 1 \implies \left(\frac{\alpha}{p}\right) = \pm 1$:

- Suppose $\left(\frac{\alpha}{p^e}\right) = 1$. Then $p \nmid \alpha$ and $\alpha \equiv \beta^2 \pmod{p}$ for some $\beta \in \mathbb{Z}$. Thus, $\left(\frac{\alpha}{p}\right) = 1$.
- Suppose $\left(\frac{\alpha}{p^e}\right) = -1$. If $p \mid \alpha$ then $\left(\frac{\alpha}{p}\right) = 0$. If $p \nmid \alpha$ then $\alpha \notin (\mathbb{Z}_p^*)^2$ and $\left(\frac{\alpha}{p}\right) = -1$.

Both by theorem (6.14).

Proving $\left(\frac{\alpha}{p^e}\right) = \pm 1 \impliedby \left(\frac{\alpha}{p}\right) = \pm 1$:
The congruence holds modulo p by applying Fermat's Little Theorem $(e-1)$ times (5.3),

$$\alpha \equiv \alpha^p \equiv \alpha^{p^2} \equiv \cdots \equiv \alpha^{p^{e-1}} \pmod{p}$$

Then $1 \equiv \alpha^{p^e} \pmod{p}$. Similarly,

$$-1 \equiv \alpha^{p^{e-1}(p-1)/2} \equiv \alpha^{(p-1)/2} \pmod{p}$$

By Euler's Criterion (6.7). ■

Chinese Remainder Map Applied to Quadratic Residues

Revisiting the Chinese Remainder Map (4.6), we show that since prime factorizations also follow the Chinese Remainder Theorem schema (a product of pairwise coprime elements), we can generate a congruence system. Also allowing us to extend such systems to residue classes.

Continued on next page...

Tip: The Chinese Remainder Theorem (CRT) is named after its origin in ancient China, where it first appeared in the work of the mathematician **Sunzi** in the 3rd century AD. In his book *Sunzi Suanjing*, he posed a problem involving finding a number that leaves specific remainders when divided by different moduli. Although the method was later formalized in modern mathematics, the name honors its roots in early Chinese mathematical texts.

Theorem 6.17: Chinese Remainder Map of Prime Factorizations

Let $n \in \mathbb{Z}^+ : 2 \nmid n$ and $n > 1$ be a product of pairwise coprime elements,

$$n = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k},$$

We map our factorization via CRT $\theta := \mathbb{Z}_n \to \mathbb{Z}_{p_1^{e_1}} \times \cdots \times \mathbb{Z}_{p_k^{e_k}}$. Then we take $\alpha \in \mathbb{Z}_n^*$ as, $\alpha := [a]_n$ and $\gcd(a, n) = 1$, allowing us to construct the map $\theta(\alpha) = \{\alpha_1, ..., \alpha_k\}$:

$$\theta(\alpha) = \begin{cases} \alpha_1 & \mod p_1^{e_1} \\ \alpha_2 & \mod p_2^{e_2} \\ \vdots & \vdots \\ \alpha_k & \mod p_k^{e_k} \end{cases}$$

Which is the system of congruences:

$$\begin{aligned} \alpha &\equiv \alpha_1 \pmod{p_1^{e_1}} \\ \alpha &\equiv \alpha_2 \pmod{p_2^{e_2}} \\ &\vdots \\ \alpha &\equiv \alpha_k \pmod{p_k^{e_k}} \end{aligned}$$

Meaning if we have some $\beta \in \mathbb{Z}_n^* : \alpha = \beta^2$, if $\theta(\beta) = \{\beta_1, ..., \beta_k\}$, then:

$$(\alpha_1, ..., \alpha_k) = \theta(\alpha) = \theta(\beta^2) = \theta(\beta)^2 = (\beta_1^2, ..., \beta_k^2)$$

Then $\alpha_i = \beta_i^2$ for each $i = 1, ..., k$. Suppose we began $\alpha_i = \beta_i^2$, for some $\beta_i \in \mathbb{Z}_{p_i^{e_i}}$. Then $(\beta_1, \ldots, \beta_k)$ is $\theta(\beta)^{-1}$. Then,

$$(\beta_1, ..., \beta_k)^2 = \theta(\beta^2) = (\beta_1^2, ..., \beta_k^2) = (\alpha_1, ..., \alpha_k) = \theta(\alpha)$$

Revealing for each $i = 1, ..., k$:

$$\alpha \in (\mathbb{Z}_n^*)^2 \iff \alpha_i \in (\mathbb{Z}_{p_i^{e_i}}^*)^2$$

Restricting θ to $(\mathbb{Z}_n^*)^2$, we get $\theta : (\mathbb{Z}_n^*)^2 \to (\mathbb{Z}_{p_1^{e_1}}^*)^2 \times \cdots \times (\mathbb{Z}_{p_k^{e_k}}^*)^2$. Then if $\alpha \in \mathbb{Z}_n^*$, there is some tuple $\{\alpha_1, ..., \alpha_k\}$ that it maps to, and vise versa, showing a bijection. Leaving us with:

$$|(\mathbb{Z}_n^*)^2| = \prod_{i=1}^{k} (\varphi(p_i^{e_i})/2) = \varphi(n)/2^k$$

By theorems (6.13) and (4.7).

From the above theorem, we relate back to theorem (6.5).

Theorem 6.18: Number of Square Roots CRT

Let $n \in \mathbb{Z}^+ : 2 \nmid n$ and $n > 1$ be a product of pairwise coprime elements, and $\alpha \in \mathbb{Z}_n^*$, where $\alpha = \beta^2$ for some $\beta \in \mathbb{Z}_n^*$ and $\theta(\beta) = \{\beta_1, ..., \beta_k\}$.

Then for some $\gamma \in \mathbb{Z}_n^*$, with $\theta(\gamma) = \{\gamma_1, ..., \gamma_k\}$, consider:

$$\begin{aligned} \gamma^2 = \beta^2 &\iff \theta(\gamma^2) = \theta(\beta^2) \\ &\iff (\gamma_1^2, \ldots, \gamma_r^2) = (\beta_1^2, \ldots, \beta_r^2) \\ &\iff (\gamma_1, \ldots, \gamma_r) = (\pm\beta_1, \ldots, \pm\beta_r). \end{aligned}$$

Therefore α has 2^k square roots, $\theta^{-1}(\pm\beta_1, ..., \pm\beta_k)$.

Square roots of -1 modulo p

Theorem 6.19: Primes are Congruent to 1 or 3 modulo 4

Let p be an odd prime. Then $p \equiv 1 \pmod 4$ or $p \equiv 3 \pmod 4$, as $p \not\equiv 0 \pmod 4$ (divisible by 4) and $p \not\equiv 2 \pmod 4$ (even but not divisible by 4).

Theorem 6.20: Quadratic Residue -1 test (mod 4)

Let p be an odd prime. Then, $\left(\frac{-1}{p}\right) = 1 \iff p \equiv 1 \pmod 4$.
I.e., -1 is a quadratic residue modulo p if and only if p is congruent to 1 modulo 4.

Proof 6.11: Quadratic Residue -1 test (mod 4)

By Euler's criterion $\left(\frac{-1}{p}\right) = 1 \iff (-1)^{(p-1)/2} \equiv 1 \pmod p$.

- If $p \equiv 1 \pmod 4$. Then, $p = 4k + 1$ for some $k \in \mathbb{Z}$. Taking 1 from both sides and dividing 2 yields, $(p-1)/2 = 2k$. Therefore, $(p-1)/2$ is even, thus $(-1)^{(p-1)/2} = 1$, as -1 raised to an even power is 1.
- If $p \equiv 3 \pmod 4$, then $p = 4k + 3$ for some $k \in \mathbb{Z}$. Then $(p-1)/2 = 2k + 1$ is odd, and $(-1)^{(p-1)/2} = -1$.

■

Theorem 6.21: Quadratic non-residue -1 test (mod 4)

Let p be a prime with $p \equiv 1 \pmod 4$, $\gamma \in \mathbb{Z}_p^* \setminus (\mathbb{Z}_p^*)^2$, and $\beta := \gamma^{(p-1)/4}$. Then $\beta^2 = -1$.

Proof 6.12: Quadratic non-residue -1 test (mod 4)

Since $p \equiv 1 \pmod 4$, we have $p = 4k + 1$ for some $k \in \mathbb{Z}$. Then $(p-1)/4 = k$ is an integer. Therefore, $\gamma^{(p-1)/4} = \beta$ is well-defined (6.17). Since γ is a non-square, then $\left(\frac{\gamma}{p}\right) = -1$, as we see by Euler's criterion:

$$\beta^2 = \gamma^{(p-1)/2} = -1$$

■

Theorem 6.22: Thue's Lemma

Let $n, b, r^*, t^* \in \mathbb{Z}$, with $0 < r^* \leq n < r^* t^*$. Then there exist $r, t \in \mathbb{Z}$ with

$$r \equiv bt \pmod n, \quad 0 < |r| < r^*, \quad \text{and} \quad 0 < |t| < t^*.$$

Meaning, $r - bt = nk : k \in \mathbb{Z}$, where r and bt differ by some multiple n. Such multiple could be enormous in magnitude. Thus we bound solutions r and t by r^* and t^* respectively.

We use this next theorem to prove Thue's Lemma.

Theorem 6.23: Pigeon Hole Principle

Let $n, m \in \mathbb{Z}^+$ with $n < m$. Then if we distribute m pigeons into n pigeonholes, there must be at least one pigeonhole with more than one pigeon.

Tip: The principle dates back to the 1830s when it was first introduced by German mathematician Peter Gustav Lejeune Dirichlet. It was originally called Dirichlet's box principle. Over time, the term "pigeonhole" became more common, referring to the analogy of pigeons and their nesting holes.

Proof 6.13: Thue's Lemma

For $\{r_i\}_{i=0}^{r^*-1}$ and $\{t_i\}_{i=1}^{t^*-1}$, we have $n_{ij}^* := i - bj$. Since $r^*t^* > n$, then by the Pigeon Hole Principle, there must be a pigeon hole (residue) with more than one pigeon (representative) as r^*t^* overlaps n. Therefore, for some $(i_1, j_1) \neq (i_2, j_2)$, we have $n_{i_1 j_1}^* = n_{i_2 j_2}^*$. We define $r := i_1 - i_2$ and $t := j_2 - j_1$, then

$$\begin{aligned} i_1 - bj_1 &\equiv i_2 - bj_2 \pmod{n} \\ i_1 - i_2 &\equiv b(j_2 - j_1) \pmod{n} \\ r &\equiv bt \pmod{n} \end{aligned}$$

Satisfying $|r| < r^*$, and $|t| < t^*$. Additionally $t \neq 0$ as that implies $r \equiv 0 \pmod{n}$, which contradicts $0 < |r|$ and $0 < |t|$. ■

Theorem 6.24: Fermat's Two Square Theorem

Let p be an odd prime. Then $p = r^2 + t^2$ for some $r, t \in \mathbb{Z}$ if and only if $p \equiv 1 \pmod{4}$.

Proof 6.14: Fermat's Two Square Theorem

Proving $p = r^2 + t^2 \implies p \equiv 1 \pmod{4}$:
The square of any integer modulo 4 is:

$$0^2 \equiv 0 \pmod{4};\ \ 1^2 \equiv 1 \pmod{4};\ \ 2^2 \equiv 0 \pmod{4};\ \ 3^2 \equiv 1 \pmod{4};$$

Our squares are either 0 or 1 modulo 4. Therefore, their possible sums are $0, 1, 2$ modulo 4, never 3. Therefore $p \equiv 3 \pmod{4}$ cannot be written as the sum of two squares.

Proving $p \equiv 1 \pmod{4} \implies p = r^2 + t^2$:
We know $p \equiv 1 \pmod{4}$, then $b^2 \equiv -1 \pmod{p}$ for some $b \in \mathbb{Z}_p^*$ (6.11). We apply Thue's Lemma (6.22): we define $n := p$ and $r^* := t^* := \lfloor \sqrt{p} \rfloor + 1$, where $\sqrt{p} \notin \mathbb{Z}$ and $\lfloor \sqrt{p} \rfloor < \sqrt{p}$. This ensures $p < r^*t^*$. Then $r^* < p$ ($\leq$ if we included 2). Thus satisfying $0 < r^* < p < r^*t^*$. Therefore, there exists $r, t \in \mathbb{Z}$,

$$r \equiv bt \pmod{p}, \quad 0 < |r| \leq \lfloor \sqrt{p} \rfloor < \sqrt{p}, \quad 0 < |t| \leq \lfloor \sqrt{p} \rfloor < \sqrt{p}$$

Then, $r^2 \equiv b^2t^2 \pmod{p}$, substituting $b^2 \equiv -1 \pmod{p}$, yields $r^2 \equiv -t^2 \pmod{p}$. Therefore, $r^2 + t^2 \equiv 0 \pmod{p}$, and $r^2 + t^2 = pk : k \in \mathbb{Z}$. Since $r^2 + t^2 < p(2)$, then $k = 1$. Thus, resulting in $p = r^2 + t^2$. ■

Theorem 6.25: Infinitely Many Primes $p \equiv 1 \pmod 4$

There are infinitely many primes $p \equiv 1 \pmod 4$.

Proof 6.15: Infinitely Many Primes $p \equiv 1 \pmod 4$

Suppose there are finitely many primes $p_1, ..., p_k \equiv 1 \pmod 4$, then all such primes are odd. Let $M := \prod_{i=1}^{k} p_i$. Then $M \equiv 1 \pmod 4$, and $M = 4n + 1$ for some arbitrary $n \in \mathbb{Z}$. We choose $N := 4M^2 + 1$, as:

$$M^2 = \left(\prod_{i=1}^{k} p_i\right)^2 = \prod_{i=1}^{k} p_i^2$$

Therefore, there are no p_i factors among M^2 or N, yielding a prime $p \notin \{p_1, ..., p_k\}$. Then,

$$\begin{aligned} N &\equiv 0 \pmod p && \text{(Given.)} \\ 4M^2 + 1 &\equiv 0 \pmod p && \text{(Substitute.)} \\ 4M^2 &\equiv -1 \pmod p && \text{(Subtract 1.)} \\ (2M)^2 &\equiv -1 \pmod p && \text{(Factor.)} \end{aligned}$$

Then -1 is also a quadratic residue modulo p, contradicting $p_1, ..., p_k$ to be only such primes. ■

Theorem 6.26: Infinitely Many Primes $p \equiv 3 \pmod 4$

There are infinitely many primes $p \equiv 3 \pmod 4$.

Proof 6.16: Infinitely Many Primes $p \equiv 3 \pmod 4$

Suppose there are finitely many primes $p_1, \ldots, p_k \equiv 3 \equiv -1 \pmod 4$. Let $M := \prod_{i=1}^{k} p_i$, which is $M = 4n - 1 : n \in \mathbb{Z}$. We choose $N := 4M - 1$. Since $N \equiv 3 \pmod 4$, then $N \equiv 0 \pmod p$ for some prime (6.19). Suppose $p \in \{p_1, \ldots, p_k\}$. Then, p must divide both N and $4M$, which implies p divides $4M - N = 1$. However, no prime can divide 1. Thus contradicting $p_1, \ldots, p_k$ to be the only such primes.

■

— 3 —

Complexity Analysis

3.1 Asymptotic Analysis

Asymptotic analysis is a method for describing the limiting behavior of functions as inputs grow infinitely.

Definition 1.1: Asymptotic

Let $f(n)$ and $g(n)$ be two functions. As n grows, if $f(n)$ grows closer to $g(n)$ never reaching, we say that "$f(n)$ is **asymptotic** to $g(n)$."

We call the point where $f(n)$ starts behaving similarly to $g(n)$ the **threshold** n_0. After this point n_0, $f(n)$ follows the same general path as $g(n)$.

Definition 1.2: Big-O: (Upper Bound)

Let f and g be functions. $f(n)$ our function of interest, and $g(n)$ our function of comparison.

Then we say $f(n) = O(g(n))$, "$f(n)$ **is big-O of** $g(n)$," if $f(n)$ grows no faster than $g(n)$, up to a constant factor. Let n_0 be our asymptotic threshold. Then, for all $n \geq n_0$,

$$0 \leq f(n) \leq c \cdot g(n)$$

Represented as the ratio $\dfrac{f(n)}{g(n)} \leq c$ for all $n \geq n_0$. Analytically we write,

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} < \infty$$

Meaning, as we chase infinity, our numerator grows slower than the denominator, bounded, never reaching infinity.

Examples:

(i.) $3n^2 + 2n + 1 = O(n^2)$

(ii.) $n^{100} = O(2^n)$

(iii.) $\log n = O(\sqrt{n})$

Proof 1.1: $\log n = O(\sqrt{n})$

We setup our ratio:

$$\lim_{n\to\infty} \frac{\log n}{\sqrt{n}}$$

Since $\log n$ and $\sqrt{n}$ grow infinitely without bound, they are of indeterminate form $\frac{\infty}{\infty}$. We apply L'Hopital's Rule, which states that taking derivatives of the numerator and denominator will yield an evaluateable limit:

$$\lim_{n\to\infty} \frac{\log n}{\sqrt{n}} = \lim_{n\to\infty} \frac{\frac{d}{dn}\log n}{\frac{d}{dn}\sqrt{n}}$$

Yielding derivatives, $\log n = \frac{1}{n}$ and $\sqrt{n} = \frac{1}{2\sqrt{n}}$. We substitute these back into our limit:

$$\lim_{n\to\infty} \frac{\frac{1}{n}}{\frac{1}{2\sqrt{n}}} = \lim_{n\to\infty} \frac{2\sqrt{n}}{n} = \lim_{n\to\infty} \frac{2}{\sqrt{n}} = 0$$

Our limit approaches 0, as we have a constant factor in the numerator, and a growing denominator. Thus, $\log n = O(\sqrt{n})$, as $0 < \infty$. ■

Definition 1.3: Big-Ω: (Lower Bound)

The symbol Ω reads "Omega." Let f and g be functions. Then $f(n) = \Omega(g(n))$ if $f(n)$ grows no slower than $g(n)$, up to a constant factor. I.e., lower bounded by $g(n)$. Let n_0 be our asymptotic threshold. Then, for all $n \geq n_0$,

$$0 \leq c \cdot g(n) \leq f(n)$$

$$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)}$$

Meaning, as we chase infinity, our numerator grows faster than the denominator, approaching 0 asymptotically.

Examples: $n! = \Omega(2^n)$; $\frac{n}{100} = \Omega(n)$; $n^{3/2} = \Omega(\sqrt{n})$; $\sqrt{n} = \Omega(\log n)$

Definition 1.4: Big Θ: (Tight Bound)

The symbol Θ reads "Theta." Let f and g be functions. Then $f(n) = \Theta(g(n))$ if $f(n)$ grows at the same rate as $g(n)$, up to a constant factor. I.e., $f(n)$ is both upper and lower bounded by $g(n)$. Let n_0 be our asymptotic threshold, and $c_1 > 0, c_2 > 0$ be some constants. Then, for all $n \geq n_0$,

$$0 \leq c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$$

$$0 < \lim_{n\to\infty} \frac{f(n)}{g(n)} < \infty$$

Meaning, as we chase infinity, our numerator grows at the same rate as the denominator.

Examples: $n^2 = \Theta(n^2)$; $2n^3 + 2n = \Theta(n^3)$; $\log n + \sqrt{n} = \Theta(\sqrt{n})$.

Definition 1.5: Little o: (Strict Upper Bound)

The symbol o reads "little-o." Let f and g be functions. Then $f(n) = o(g(n))$ if $f(n)$ grows strictly slower than $g(n)$, meaning $f(n)$ becomes insignificant compared to $g(n)$ as n grows large. Let n_0 be our asymptotic threshold. Then, for all $n \geq n_0$,

$$0 \leq f(n) < c \cdot g(n)$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

Meaning, as we chase infinity, the ratio of $f(n)$ to $g(n)$ shrinks to zero.

Examples: $n = o(n^2)$; $\log n = o(n)$; $n^{0.5} = o(n)$.

Definition 1.6: Little ω: (Strict Lower Bound)

The symbol ω reads "little-omega." Let f and g be functions. Then $f(n) = \omega(g(n))$ if $f(n)$ grows strictly faster than $g(n)$, meaning $g(n)$ becomes insignificant compared to $f(n)$ as n grows large. Let n_0 be our asymptotic threshold. Then, for all $n \geq n_0$,

$$0 \leq c \cdot g(n) < f(n)$$

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = 0$$

Meaning, as we chase infinity, the ratio of $g(n)$ to $f(n)$ shrinks to zero.

Examples: $n^2 = \omega(n)$; $n = \omega(\log n)$.

Definition 1.7: Asymptotic Equality ($\sim$)

The symbol $\sim$ reads "asymptotic equality." Let f and g be functions. Then $f(n) \sim g(n)$ if, as $n \to \infty$, the ratio of $f(n)$ to $g(n)$ approaches 1. I.e., the two functions grow at the same rate asymptotically. Formally,

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 1$$

Meaning, as n grows large, the two functions become approximately equal.

Examples: $n + 100 \sim n$, $\log(n^2) \sim 2\log n$.

Tip: To review:

- **Big-O:** $f(n) < g(n)$ (Upper Bound); $f(n)$ grows no faster than $g(n)$.
- **Big-Ω:** $f(n) > g(n)$ (Lower Bound); $f(n)$ grows no slower than $g(n)$.
- **Big-Θ:** $f(n) = g(n)$ (Tight Bound); $f(n)$ grows at the same rate as $g(n)$.
- **Little-o:** $f(n) < g(n)$ (Strict Upper Bound); $f(n)$ grows strictly slower than $g(n)$.
- **Little-ω:** $f(n) > g(n)$ (Strict Lower Bound); $f(n)$ grows strictly faster than $g(n)$.
- **Asymptotic Equality:** $f(n) \sim g(n)$; $f(n)$ grows at the same rate as $g(n)$.

Theorem 1.1: Types of Asymptotic Behavior

The following are common relationships between different types of functions and their asymptotic growth rates:

- **Polynomials.** Let $f(n) = a_0 + a_1 n + \cdots + a_d n^d$ with $a_d > 0$. Then, $f(n)$ is $\Theta(n^d)$.
  E.e., $3n^2 + 2n + 1$ is $\Theta(n^2)$.
- **Logarithms.** $\Theta(\log_a n)$ is $\Theta(\log_b n)$ for any constants $a, b > 0$. That is, logarithmic functions in different bases have the same growth rate.
  E.g., $\log_2 n$ is $\Theta(\log_3 n)$.
- **Logarithms and Polynomials.** For every $d > 0$, $\log n$ is $O(n^d)$. This indicates that logarithms grow slower than any polynomial.
  E.g., $\log n$ is $O(n^2)$.
- **Exponentials and Polynomials.** For every $r > 1$ and every $d > 0$, n^d is $O(r^n)$. This means that exponentials grow faster than any polynomial.
  E.e., n^2 is $O(2^n)$.

3.2 Evaluating Algorithms

When analyzing algorithms, we are interested in two primary factors: time and space complexity.

Definition 2.1: Time Complexity

The **time complexity** of an algorithm is the amount of time it takes to run as a function of the input size. We use asymptotic notation to describe the time complexity of an algorithm.

Definition 2.2: Space Complexity

The **space complexity** of an algorithm is the amount of memory it uses to store inputs and subsequent variables during the algorithm's execution. We use asymptotic notation to describe the space complexity of an algorithm.

Below is an example of a function and its time and space complexity analysis.

Function 2.1: Arithmetic Series - `Fun1(`A`)`

Computes a result based on a length-n array of integers:

Input: A length-n array of integers.
Output: An integer p computed from the array elements.

```
1 Function Fun1(A):
2 |  p ← 0;
3 |  for i ← 1 to n − 1 do
4 |  |  for j ← i + 1 to n do
5 |  |  |  p ← p + A[i] · A[j];
6 |  |  end
7 |  end
8 return p
```

Time Complexity: For $f(n) :=$ `Fun1(`A`)`, $f(n) = \frac{n^2}{2} = O(n^2)$. This is because the function has a nested loop structure, where the inner for-loop runs $n - i$ times, and the outer for-loop runs $n - 1$ times. Thus, the total number of iterations is $\sum_{i=1}^{n-1} n - i = \frac{n^2}{2}$.

Space Complexity: We yield $O(n)$ for storing an array of length n. The variable p is $O(1)$ (constant), as it is a single integer. Hence, $f(n) = n + 1 = O(n)$.

Additional Example: Let $f(n, m) = n^2m + m^3 + nm^3$. Then, $f(n, m) = O(n^2m + m^3)$. This is because both n and m must be accounted for. Our largest n term is n^2m, and our largest m term is m^3 both dominate the expression. Thus, $f(n, m) = O(n^2m + m^3)$.

3.3 Computers & Number Base Systems

The following section introduces the foundation of all computer science.

Definition 3.1: Turing Machine

A **Turing Machine** is a theoretical computational model used to describe the capabilities of a general-purpose **computer**. It consists of an infinite tape (memory) and a read/write head processing symbols on the tape, one at a time, according to a set of predefined rules. The machine moves left or right, reading or writing symbols, and changing states based on what it reads.

The machine **halts** once it reaches a final state or continues indefinitely. Serving as a flexible, **higher-order function** (a function which receives functions).

Definition 3.2: Von Neumann Architecture

Modern computers operate on a model known as the **Von Neumann architecture**, which consists of three primary components:

1. **Memory**: Stores data and instructions as sequences of bits.
2. **Arithmetic and Logic Unit (ALU)**: Executes operations such as addition, subtraction, multiplication, and division on numbers stored in memory.
3. **Control Unit**: Directs the execution of instructions and manages the flow of data between memory and the ALU.

Where numbers are stored in memory cells, each cell holding an integer value represented in a fixed base, typically $B = 2$, meaning **binary**. Where each digit is less than the base B. We represent integers in memory as:

$$a = \sum_{i=0}^{k-1} a_i B^i$$

where a_i represents the individual digits, and B is the base. For large integers, computations may require manipulating several memory cells to store the full number.

Tip: Alan Turing (1912-1954) was a British mathematician and logician. In 1936, he introduced the concept of the Turing machine, laying the groundwork for theory of computation. During World War II Turing broke the German Enigma encrypted code shifting the tides of the war. He after worked on early computers and artificial intelligence, proposing the "Turing Test", defining machine intelligence. Tragically, Turing's life was cut short by personal and professional persecution against his homosexuality, dying at 41.

Tip: John von Neumann (1903-1957) was a Hungarian-American mathematician and polymath who made fundamental contributing to, mathematics, physics, economics, and computer science. He is best known for developing the "*Von Neumann Architecture.*"

Example: Consider the integer $a = 13$, and let us represent it in base $B = 2$ (binary). We can express this number as a sum of powers of 2, corresponding to the binary representation of 13:

$$a = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 8 + 4 + 0 + 1 = 13$$

In binary, this is represented as the sequence of digits: $a = (1101)_2$. Here, the coefficients $a_3 = 1$, $a_2 = 1$, $a_1 = 0$, and $a_0 = 1$ correspond to the binary digits of 13.
Similarly, if we want to represent $a = 45$ in base $B = 10$ **(decimal)**, we write:

$$a = 4 \cdot 10^1 + 5 \cdot 10^0 = 40 + 5 = 45$$

In this case, the coefficients $a_1 = 4$ and $a_0 = 5$ correspond to the decimal digits of 45.

Definition 3.3: Hexadecimal

Hexadecimal base $B = 16$, using digits $0 - 9$ and the letters $A - F$, where $A = 10$, $B = 11$, $C = 12$, $D = 13$, $E = 14$, and $F = 15$. Hexadecimal is commonly used in computing due to its compact representation of binary data. For example, a **byte** (8 bits) can be represented as two hexadecimal digits, simplifying the display of binary data.

Theorem 3.1: Base $2 \leftrightarrow 16$ Conversion

Let bases $B := 2$ (binary) and $H := 16$ (hexadecimal). At a high-level:

Binary to Hexadecimal:

1. Group B digits in sets of 4, right to left. **Pad** leftmost group with 0's if necessary for a full group.
2. Compute each group, replacing the result with their H digit.
3. Finally, combine each H group.

Hexadecimal to Binary:

1. Convert each H digit into a 4 bit B group.
2. Finally, combine all B groups.

Additionally we may also trim any leading 0's.

Example:

- Binary to Hexadecimal:

$$101101111010_2 \quad \Rightarrow \quad \text{Group as ([1011] [0111] [1010])} \quad \Rightarrow \quad B7A_{16}$$

- Hexadecimal to Binary:

$$3F5_{16} \quad \Rightarrow \quad [0011]\ [1111]\ [0101]_2 \Rightarrow \quad 1111110101_2$$

The following definition is for completeness: applications of such a base are currently uncommon.

Definition 3.4: Unary

Unary, base $B = 1$. A system where each number is represented by a sequence of B symbols. Where the number n is represented by n symbols. Often used in theoretical computer science to prove the existence of computable functions.

Example: The number 5 in unary is represented by 5 symbols: $5 = \texttt{IIIII}$ or $2 = \texttt{II}$. There is no concept of 0, the absence of symbols represents 0.

Definition 3.5: Most & Least Significant Bit

In a binary number, the **most significant bit (MSB)** is the leftmost bit. The **least significant bit (LSB)** is the rightmost bit.

Theorem 3.2: Adding Binary

We may use the add and carry method alike decimal addition: **Binary Addition Rules:**

- $0 + 0 = 0$
- $0 + 1 = 1$
- $1 + 0 = 1$
- $1 + 1 = 0$ (add 1 to the next digit (left))

We call the last step, **carry**, as we carry our overflow to the next digit.

Example: Adding $0010\ 0011\ 0100_2$ and 0100_2:

$$\begin{array}{r} 0010\ 0011\ 0100 \\ + \qquad\quad 0100 \\ \hline 0010\ 0011\ 1000 \end{array}$$

Where the $0100 + 0100 = 1000$ as we carry the an overflow of 1.

Theorem 3.3: Signed Binary Numbers - Two's Complement

In a **two's complement system**, an n-bit signed (positive or negative) binary number can represent values in the range $[-2^{n-1}, 2^{n-1} - 1]$. Then by most significant bit (MSB):

- If MSB is 0, the number is positive;
- If MSB is 1, the number is negative.

Conversion to Two's Complement :

1. Take an unsigned binary number and invert all bits, turning 0's to 1's and 1's to 0's.
2. Finally add 1 to the least significant bit.

Example: Converting -5 into a 4-bit two's complement:

$$\begin{array}{rl} 5 \to 0101 & \text{(binary for 5)} \\ \quad 1010 & \text{(inverted)} \\ \quad 1011 & \text{(add 1)} \end{array}$$

Thus, -5 is represented as 1011 in 4-bits under two's complement.

3.4 Computing Large Numbers

In this section we discuss algorithms for computing large numbers, but first we define algorithmically, addition, subtraction, multiplication, division, and modula.

Definition 4.1: Wordsize

Our machine has a fixed **wordsize**, which is how much each memory cell can hold. Systems like 32-bit or 64-bit can hold 2^{32} (≈ 4.3 billion) or 2^{64} (≈ 18.4 quintillion) bits respectively.

We say the ALU can performs arithmetic operations at $O(1)$ time, within wordsize. Operations beyond this size we deem **large numbers**.

The game we play in the following algorithms is to compute large integers without exceeding wordsize. Moving forward, we assume our machine is a typical 64-bit system.

Definition 4.2: Computer Integer Division

Our ALU only returns the quotient after division. We denote the quotient as $\lfloor a/b \rfloor : a, b \in \mathbb{Z}$.

Our first hurdle is long division as , which will set up long addition and subtraction for success.

Function 4.1: Division with Remainder (Outline) - *QuoRem()*

For base B integers, let dividend $a = (a_{k-1} \cdots a_0)_B$ and divisor $b = (b_{\ell-1} \cdots b_0)_B$ be unsigned, with $k \geq 1$, $\ell \geq 1$, ensuring $0 \leq b \leq a$, and $b_{\ell-1} \neq 0$, ensuring $b > 0$.

We compute q and r such that, $a = bq + r$ and $0 \leq r < b$. Assume $k \geq \ell$; otherwise, $a < b$. We set $q \leftarrow 0$ and $r \leftarrow a$. Then quotient $q = (q_{m-1} \cdots q_0)_B$ where $m := k - \ell + 1$.

Note: Our general strategy below won work as $\lfloor r/B^i b \rfloor$ exceeds wordsize quickly. We finalize our strategy in Function (4.2).

Input: a, b
Output: q, r

```
1 Function QuoRem(a, b):
2     r ← a;
3     q ← {0_{m-1} ··· 0};
4     for i ← m − 1 down to 0 do
5         q_i ← ⌊r/B^i b⌋;
6         r ← r − B^i · q_i b;
7     end
8     return (q, r);
9 return
```

Time Complexity: $O(k-\ell)$, as we iterate $m := k-\ell+1$ times, ignoring our constant factor.
Space Complexity: $O(k+\ell)$, as we store inputs a, b.

***QuoRem()* Lines 5 and 6** : similar to grade-school long division: compute the quotient, subtract the product of the divisor and quotient from the dividend, repeating until the dividend is less than the divisor. **Examples:** let $a = \{12, 5, 17, 40, 89\}$, $b = \{4, 2, 3, 9, 10\}$ respectively, and $B = 10$,

$$\begin{array}{r|l} & 3 \\ 4 & 12 \\ & 12 \\ \hline & 0 \end{array} \qquad \begin{array}{r|l} & 2 \\ 2 & 5 \\ & 4 \\ \hline & 1 \end{array} \qquad \begin{array}{r|l} & 5 \\ 3 & 17 \\ & 15 \\ \hline & 2 \end{array} \qquad \begin{array}{r|l} & 4 \\ 9 & 40 \\ & 36 \\ \hline & 4 \end{array} \qquad \begin{array}{r|l} & 8 \\ 10 & 89 \\ & 80 \\ \hline & 9 \end{array}$$

Take $a = 17$ and $b = 3$: 3 fits into 17 five times, which is 15. 17 take away 15 is 2, our remainder.

Proof 4.1: ($q_i \leftarrow \lfloor \frac{r}{B^i b} \rfloor$) and ($r \leftarrow r - B^i \cdot q_i b$)

Let r be the current remainder in the division process. We aim to compute q_i, the digit of the quotient corresponding to the power B^i in the base-B representation. The divisor is b, and the base is B.

For Example: $a = 5$ and $b = 2$ in base $B = 10$.
Dividend 5, represented by each B^i.

1. **Estimate Quotient Digit (line 5)**
We estimate how many times the scaled divisor $B^i \cdot b$ fits into the remainder r in integer division gives us:

$$q_i \leftarrow \left\lfloor \frac{r}{B^i b} \right\rfloor \quad \text{e.g.,} \quad q_i \leftarrow \left\lfloor \frac{5}{2^1 \cdot 2} \right\rfloor = \left\lfloor \frac{5}{4} \right\rfloor = 1$$

This operation selects the largest integer q_i such that:

$$q_i \cdot B^i \cdot b \leq r \quad \text{e.g.,} \quad 1 \cdot 2^1 \cdot 2 = 4 \leq 5$$

2. **Update Remainder (line 6)**

$$r \leftarrow r - B^i \cdot q_i b \quad \text{e.g.,} \quad r \leftarrow 5 - 2^1 \cdot (1)(2) = 5 - 4 = 1. \text{ Recall, } \begin{array}{r} 2 \\ 2\overline{)5} \\ \underline{4} \\ 1 \end{array}$$

3. **Correctness**
Since q_i was chosen as the largest integer such that $0 \leq r < B^i \cdot b$, through each iteration.

■

Though we need to adapt this algorithm for binary numbers. Consider the case $\ell = 1$ (1 binary digit) and $B = 2$:

Theorem 4.1: 2's Powers - Division Algorithm (Part 1)

Let $x, y \in \mathbb{Z}$, base 10, be represented in the division algorithm as 2's powers such that:

$$0 \leq x = x'2^n + s \quad \text{and} \quad 0 < y = y'2^n$$

for some $n, s, x', y' \in \mathbb{Z}$, base 10. Where $n \geq 0$ and $0 \leq s < 2^n$. Then,

$$\left\lfloor \frac{x}{y} \right\rfloor = \left\lfloor \frac{x'}{y'} \right\rfloor.$$

This will help us working in binary, base 2, as for some integer $a = \sum_{i=0}^{k-1} a_i 2^i$, we can estimate a as $a = a'2^n + s$ for some $n, s, a' \in \mathbb{Z}$.

Proof 4.2: 2's Powers - Division Algorithm (Part 1)

We have

$$\frac{x}{y} = \frac{x'2^n + s}{y'2^n} = \frac{x'}{y'} + \frac{s}{y'2^n} \geq \frac{x'}{y'}.$$

It follows immediately that $\left\lfloor \frac{x}{y} \right\rfloor \geq \left\lfloor \frac{x'}{y'} \right\rfloor$. Then,

$$\frac{x}{y} = \frac{x'}{y'} + \frac{s}{y'2^n} < \frac{x'}{y'} + \frac{1}{y'} < \left\lfloor \frac{x'}{y'} \right\rfloor + 1.$$

Thus, $\left\lfloor \frac{x'}{y'} \right\rfloor \leq \frac{x}{y} < \left\lfloor \frac{x'}{y'} \right\rfloor + 1$. Hence, $\left\lfloor \frac{x}{y} \right\rfloor = \left\lfloor \frac{x'}{y'} \right\rfloor$. ■

Now we consider general cases $\ell \geq 1$ and $B = 2$.

Theorem 4.2: 2's Powers - Division Algorithm (Part 2)

Let $x, y \in \mathbb{Z}$, base 10, be represented in the division algorithm as 2's powers such that:

$$0 \leq x = x'2^n + s \quad \text{and} \quad 0 < y = y'2^n + t$$

for some $n, s, t, x', y' \in \mathbb{Z}$, base 10. Where $n \geq 0$, $0 \leq s < 2^n$, and $0 \leq t < 2^n$. Then,

$$\left\lfloor \frac{x}{y} \right\rfloor \leq \left\lfloor \frac{x'}{y'} \right\rfloor \leq \left\lfloor \frac{x}{y} \right\rfloor + 2$$

Proof 4.3: 2's Powers - Division Algorithm (Part 2)

We have $\frac{x}{y} = \frac{x'2^n + s}{y'2^n + t}$, then $\left\lfloor \frac{x}{y} \right\rfloor \leq \left\lfloor \frac{x}{y'2^n} \right\rfloor$. Then by part 1 we substitute $y'2^n$ for y, yielding $\left\lfloor \frac{x}{y'2^n} \right\rfloor = \left\lfloor \frac{x'}{y'} \right\rfloor$, proving the first inequality $\left(\left\lfloor \frac{x}{y} \right\rfloor \leq \left\lfloor \frac{x'}{y'} \right\rfloor \right)$. Then we note $\frac{x}{y} \geq \frac{x'}{y'+1}$, then by cross multiplication: $x(y'+1) \geq y(x')$, and $x'y - xy' - x \leq 0$. Likewise, $\frac{2y'}{\geq} \frac{x}{y}$, implies $2yy' - x \geq 0$, yielding:

$$2yy' - x \geq 0 \geq x'y - xy' - x \Longrightarrow \frac{x}{y} \geq \frac{x'}{y'-2}$$

Thus, $\left\lfloor \frac{x}{y} \right\rfloor \geq \left\lfloor \frac{x'}{y'} \right\rfloor - 2$, proving the second inequality $\left(\left\lfloor \frac{x'}{y'} \right\rfloor \leq \left\lfloor \frac{x}{y} \right\rfloor + 2 \right)$.

■

Example: Consider the following powers of 2 of form $x = 2^n + s$:

$$3 = 2 + 1 = 0000\ 0011_2 \quad (1)$$
$$6 = 4 + 2 = 0000\ 0110_2 \quad (2)$$
$$12 = 8 + 4 = 0000\ 1100_2 \quad (3)$$
$$24 = 16 + 8 = 0001\ 1000_2 \quad (4)$$
$$48 = 32 + 16 = 0011\ 0000_2 \quad (5)$$
$$96 = 64 + 32 = 0110\ 0000_2 \quad (6)$$
$$192 = 128 + 64 = 1100\ 0000_2 \quad (7)$$

Notice that as we increase the power of 2, the number of bits shift left towards a higher-order bit. We apply this theorem directly to shifting bits to yield instant calculations.

Theorem 4.3: Binary Bit Shifting (Powers of 2)

Let x_2 be a base 2 unsigned integer. Then,
Left Shift by k: $x \ll k := x \cdot 2^k$
Right Shift by k: $x \gg k = \lfloor x/2^k \rfloor$
Remainder: bits pushed out after right shift.

Example: Observe, 16 = 10000 4 trailing zeros, 8 = 1000 3 trailing zeros, we shift by 4 and 3 respectively:

- Instead of $3 \cdot 16$ in base 10, we can $3 \ll 4 = 48$, as $3 \cdot 2^4 = 48$.
- Conversely, Instead of 48/16 in base 10, $48 \gg 4 = 3$, as $\lfloor 48/2^4 \rfloor = 3$.
- Catching the remainder: say we have 37/8 base 10, then,

$$37 = 100101_2 \quad \text{and} \quad 8 = 1000_2$$

$$37 \gg 3 = 4 \text{ remainder } 5,$$

  as $[100101] \gg 3 = [000100]101$, where 101_2 is our remainder 5_{10}.

Next we revisit our division algorithm, employing bit shifting to optimize our calculations.

Tip: The Z3, developed by Konrad Zuse in 1941 in Berlin, Germany, was the first programmable binary computer, using electromechanical relays processing numbers on 22-bit floating-point arithmetic. Unfortunately, the machine was destroyed in a Allied bombing raid during World War II.

The first fully electronic binary computers came later, using bits and bytes to represent data. Machines like ENIAC (1945) in the U.S. operated on a decimal system initially but were pivotal in transitioning to binary-based computing, which became the standard.

Optimizing our Outline Function (4.1) with bit shifting we have:

Function 4.2: Division with Remainder (Binary Version) - *QuoRem()*

For binary integers, let dividend $a = (a_{k-1} \cdots a_0)_2$ and divisor $b = (b_{\ell-1} \cdots b_0)_2$ be unsigned, with $k \geq 1$, $\ell \geq 1$, ensuring $0 \leq b \leq a$, and $b_{\ell-1} \neq 0$, ensuring $b > 0$.

We compute q and r such that, $a = bq + r$ and $0 \leq r < b$. Assume $k \geq \ell$; otherwise, $a < b$. We set $q \leftarrow 0$ and $r \leftarrow a$. Then quotient $q = (q_{m-1} \cdots q_0)_2$ where $m := k - \ell + 1$.

Input: a, b (binary integers)
Output: q, r (quotient and remainder in binary)

1 **Function** *QuoRem*(a, b)**:**
2 $\quad r \leftarrow a$;
3 $\quad q \leftarrow \{0_{m-1} \cdots 0\}$;
4 $\quad$ **for** $i \leftarrow m - 1$ ***down to*** 0 **do**
5 $\quad\quad q_i \leftarrow \left\lfloor \dfrac{r}{b \ll i} \right\rfloor$;
6 $\quad\quad r \leftarrow r - (q_i \cdot (b \ll i))$;
7 $\quad$ **end**
8 $\quad$ **return** (q, r);
9 **return**

Time Complexity: $O(k-\ell)$, as we iterate $m := k-\ell+1$ times, ignoring our constant factor.
Space Complexity: $O(k+\ell)$, as we store inputs a, b.

Example: Let $a = 47_10 = 101111_2$ and $b = 5_10 = 101_2$, we run *QuoRem*(a, b):

- **Step 1: Initialization**
  - Set $r \leftarrow 47 = 101111_2$, $q = 0$, $i = (k - \ell + 1) - 1 = (6 - 3 + 1) - 1 = 3$.
- **Step 2: Iteration and Bit-Shifting (lines 5 and 6 of the algorithm)**
  - For $i = 3$, we shift $5 = 101_2$ by 3 bits to get $101000_2 = 40_{10}$.
  - Now compute $q_3 \leftarrow \left\lfloor \frac{r}{b \ll 3} \right\rfloor = \left\lfloor \frac{47}{40} \right\rfloor = 1$.
  - Update $r \leftarrow r - (q_3 \cdot (b \ll 3)) = 47 - 40 = 7$.
  - End of the iteration $q = 1000_2 = 8_{10}$.
- **Step 3: Second Iteration**
  - For $i = 2$, shift $5 = 101_2$ by 2 bits to get $10100_2 = 20_{10}$.
  - Compute $q_2 \leftarrow \left\lfloor \frac{r}{b \ll 2} \right\rfloor = \left\lfloor \frac{7}{20} \right\rfloor = 0$.
  - No update for r or q, so q remains 1000_2.

- **Step 4: Third Iteration**
  - For $i = 1$, shift $5 = 101_2$ by 1 bit to get $1010_2 = 10_{10}$.
  - Compute $q_1 \leftarrow \left\lfloor \frac{r}{b \ll 1} \right\rfloor = \left\lfloor \frac{7}{10} \right\rfloor = 0$.
  - No update for r or q, so q remains 1000_2.
- **Step 5: Final Iteration**
  - For $i = 0$, shift $5 = 101_2$ by 0 bits to get $101_2 = 5_{10}$.
  - Compute $q_0 \leftarrow \left\lfloor \frac{r}{b \ll 0} \right\rfloor = \left\lfloor \frac{7}{5} \right\rfloor = 1$.
  - Update $r \leftarrow r - (q_0 \cdot (b \ll 0)) = 7 - 5 = 2$.
  - Update $q \leftarrow 1001_2 = 9_{10}$.
- **Step 6: Final Result**
  - The quotient $q = 1001_2 = 9_{10}$.
  - The remainder $r = 2$.

Thus, $47 \div 5$ gives quotient 9 and remainder 2.
Notice how this exactly matches the long division algorithm for decimal numbers.

$$\begin{array}{r} 9 \\ 5\overline{)47} \\ \underline{45} \\ 2 \end{array}.$$

We summarize the above example as, "How many times does 101_2 fit into 101111_2?"

1. Does $5 \ll 3$ fit into 101000_2? It fits! $q = 1000_2$, $r = 101111_2 - 101000_2$.
2. Does $5 \ll 2$ fit into 10100_2? no fits! $q = 1000_2$, $r = 0111_2$.
3. Does $5 \ll 1$ fit into 10100_2? no fits! $q = 1000_2$, $r = 0111_2$.
4. Does $5 \ll 0$ fit into 10100_2? It fits! $q = 1001_2$, $r = 0111_2 - 0101$.
5. Return $q = 1001_2 = 9_{10}$, $r = 0010 = 2_{10}$

Tip: Bit shifting is a low-level operation frequently used in assembly language, C, and other systems programming languages. It enables efficient manipulation of binary data, often used for tasks like multiplying or dividing by powers of 2. In assembly, shifting is closely tied to hardware-level optimizations, while in C, operators like `<<` (left shift) and `>>` (right shift) perform shifts directly on integers. Other forms of bit manipulation, like masking and rotating bits, often complement shifts in optimizing algorithms.

Long addition, subtraction, and multiplication follow directly.

Function 4.3: Addition of Base-B Integers - *Add()*

Let $a = (a_{k-1} \cdots a_0)_B$ and $b = (b_{\ell-1} \cdots b_0)_B$ be unsigned integers, where $k \geq \ell \geq 1$. We compute $c := a + b$ where the result $c = (c_k c_{k-1} \cdots c_0)_B$ is of length $k + 1$, assuming $k \geq \ell$. If $k < \ell$, swap a and b. This algorithm computes the base-B representation of $a + b$.

Input: a, b (base-B integers)
Output: $c = (c_k \cdots c_0)_B$ (sum of $a + b$)

```
1  Function Add(a, b):
2      carry ← 0;
3      for i ← 0 to ℓ − 1 do
4          tmp ← a_i + b_i + carry;
5          (carry, c_i) ← QuoRem(tmp, B);
6      end
7      for i ← ℓ to k − 1 do
8          tmp ← a_i + carry;
9          (carry, c_i) ← QuoRem(tmp, B);
10     end
11     c_k ← carry;
12     return c = (c_k ··· c_0)_B;
13 return
```

Note: $0 \leq carry \leq 1$ and $0 \leq temp \leq (2B - 1)$.
Time Complexity: $O(k)$, since we iterate over the digits of the integers, with $k \geq \ell$.
Space Complexity: $O(k + l)$, even though $c = k + 1$, constants are negligible approaching infinity.

For page-space on the subtraction algorithm, we interject with the following observation:

Function 4.4: Binary Length of a Number - $||a||$

The binary length of an integer a_{10} in binary representation, is given by:

$$||a|| := \begin{cases} \lfloor \log_2 |a| \rfloor + 1 & \text{if } a \neq 0, \\ 1 & \text{if } a = 0, \end{cases}$$

as $\lfloor \log_2 |a| \rfloor + 1$ correlates to the highest power of 2 required to represent a.

Function 4.5: Subtraction of Base-B Integers - *Sub()*

Let $a = (a_{k-1} \cdots a_0)_B$ and $b = (b_{\ell-1} \cdots b_0)_B$ be unsigned integers, where $k \geq \ell \geq 1$. We compute $c := a - b$ where the result $c = (c_{k-1} \cdots c_0)_B$ is of length k, assuming $k \geq \ell$. If $k < \ell$, swap a and b. This algorithm computes the base-B representation of $a - b$.

Input: a, b (base-B integers)
Output: $c = (c_{k-1} \cdots c_0)_B$ (difference of $a - b$)

```
1  Function Sub(a, b):
2      carry ← 0;
3      for i ← 0 to ℓ − 1 do
4          tmp ← a_i − b_i + carry;
5          (carry, c_i) ← QuoRem(tmp, B);
6      end
7      for i ← ℓ to k − 1 do
8          tmp ← a_i + carry;
9          (carry, c_i) ← QuoRem(tmp, B);
10     end
11     if carry = −1 then
12         Handle negative case using another subtraction routine for b − a;
13     end
14     c_k ← carry;
15     return c = (c_{k−1} ··· c_0)_B;
16 return
```

Note: In every loop iteration, the carry is either 0 or -1, and tmp lies between $-B$ and $B-1$. If $a \geq b$, then there is no carry out of the last loop, and $c_k = 0$. Otherwise, the carry $c_k = -1$, and the subtraction may need to be computed again.
Time Complexity: $O(k)$, as we iterate over the digits of the integers, with $k \geq \ell$.
Space Complexity: $O(k)$, as we store the digits of a, b, and c.

Definition 4.3: Most & Least Significant Bit (MSB & LSB)

The left most bit in a binary number a is the **most significant bit** and the first bit, the **least significant bit.**

Example: Consider this byte (8 bits), $[1111\ 1110]_2$, the MSB = 1 and the LSB = 0.

Function 4.6: Multiplication of Base-B Integers - *Mul()*

Let $a = (a_{k-1} \cdots a_0)_B$ and $b = (b_{\ell-1} \cdots b_0)_B$ be unsigned integers, where $k \geq 1$ and $\ell \geq 1$. The product $c := a \cdot b$ is of the form $(c_{k+\ell-1} \cdots c_0)_B$, and may be computed in time $O(k\ell)$ as follows:

Input: a, b (base-B integers)
Output: $c = (c_{k+\ell-1} \cdots c_0)_B$ (product of $a \cdot b$)

```
1  Function Mul(a, b):
2      for i ← 0 to k + ℓ − 1 do
3          c_i ← 0;
4      end
5      for i ← 0 to k − 1 do
6          carry ← 0;
7          for j ← 0 to ℓ − 1 do
8              tmp ← a_i · b_j + c_{i+j} + carry;
9              (carry, c_{i+j}) ← QuoRem(tmp, B);
10         end
11         c_{i+ℓ} ← carry;
12     end
13     return c = (c_{k+ℓ−1} ··· c_0)_B;
14 return
```

Note: At every step, the value of *carry* lies between 0 and $B - 1$, and the value of *tmp* lies between 0 and $B^2 - 1$.
Time Complexity: $O(k\ell)$, since the algorithm involves $k \cdot \ell$ multiplications.
Space Complexity: $O(k + \ell)$, since we store the digits of a, b, and c.

Tip:

In low-level languages like Assembly and C, memory management is not abstracted away as it is in higher-level languages. Programmers need to directly manage memory, ensuring efficient use of resources. Bitwise operations, such as shifting bits left or right, allow precise control over data manipulation, providing speedups in calculations like multiplication or division by powers of two. These operations are crucial in writing efficient code for systems with limited resources, such as embedded systems.

However, poor memory management, such as neglecting bounds checking or failing to free unused memory, can introduce security risks. Buffer overflows, where more data is written than a buffer can hold, allow attackers to overwrite memory and potentially inject malicious code. Techniques such as stack canaries and address space layout randomization (ASLR) are modern defenses against such attacks, but understanding binary operations remains essential for both writing optimized code and protecting against security vulnerabilities in lower-level programming.

Function 4.7: Decimal to Binary Conversion - *DecToBin()*

This function converts a decimal number n into its binary equivalent by repeatedly dividing the decimal number by 2 and recording the remainders.

Input: n (a decimal number)
Output: b (binary representation of n)

```
1 Function DecToBin(n):
2 |   b ← empty string;
3 |   while n > 0 do
4 |   |   r ← n mod 2;
5 |   |   n ← ⌊n/2⌋;
6 |   |   b ← r + b;
7 |   end
8 |   return b;
9 return
```

Time Complexity: $O(\log n)$, as the number of iterations is proportional to the number of bits in n.
Space Complexity: $O(n)$, storing our input n.

Example: Converting 89 to binary given the above function:

$$\begin{aligned}
89_{10} \div 2 &= 44 \quad \text{rem } 1, \longleftarrow \textbf{ LSB} \\
44_{10} \div 2 &= 22 \quad \text{rem } 0, \\
22_{10} \div 2 &= 11 \quad \text{rem } 0, \\
11_{10} \div 2 &= 5 \quad \text{rem } 1, \\
5_{10} \div 2 &= 2 \quad \text{rem } 1, \\
2_{10} \div 2 &= 1 \quad \text{rem } 0, \\
1_{10} \div 2 &= 0 \quad \text{rem } 1. \longleftarrow \textbf{ MSB}
\end{aligned}$$

Thus, $89_{10} = 1011001_2$.

Theorem 4.4: Computational Complexity of Basic Arithmetic Operations

Let a and b be arbitrary integers.

(i) We can compute $a \pm b$ in time $O(\text{len}(a) + \text{len}(b))$.

(ii) We can compute $a \cdot b$ in time $O(\text{len}(a) \cdot \text{len}(b))$.

(iii) If $b \neq 0$, we can compute the quotient $q := \left\lfloor \frac{a}{b} \right\rfloor$ and the remainder $r := a \mod b$ in time $O(\text{len}(b) \cdot \text{len}(q))$.

3.5 Computing in $\mathbb{Z}_n$

Recall the Least Residue definition (2.3). To stress, if $a \in \mathbb{Z}$, and $\in \mathbb{Z}_n := \{0, \ldots, n-1\}$ Then a is the least residue. Moreover, $\alpha := [a]_n$, where α or a are used interchangeably, as they refer to the same element in $\mathbb{Z}_n$. Let $\beta := [b]_n\mathbb{Z}_n$, then $\alpha + \beta = [a+b]_n = a + b$ if and only if a and b are the least residues.

Function 5.1: Least Residue Representation - *rep()*

Given an integer $a \in \mathbb{Z}_n$, ***rep(a)*** refers to the least residue representation of a modulo n.

Theorem 5.1: Arithmetic Operations in $\mathbb{Z}_n$

Let $\alpha, \beta \in \mathbb{Z}_n$, where n is a modulus. We define the following operations in terms of their least residue representations *rep(·)*:

- **Addition:** To compute $\alpha + \beta$ in $\mathbb{Z}_n$, we first calculate the integer sum $rep(\alpha) + rep(\beta)$, then subtract n if the result is greater than or equal to n.
- **Subtraction:** To compute $\alpha - \beta$ in $\mathbb{Z}_n$, we first calculate the integer difference $rep(\alpha) - rep(\beta)$, adding n if the result is negative.
- **Multiplication:** To compute $\alpha \cdot \beta$ in $\mathbb{Z}_n$, we calculate $rep(\alpha) \cdot rep(\beta) \mod n$ using integer multiplication followed by a division with remainder.

These operations can be performed in time complexities:

- Addition and Subtraction: $O(||n||)$
- Multiplication, Division, and mod: $O(||n^2||)$

Definition 5.1: Binary Inputs and Memory Allocation

Say we want an array of 4 integers, all 4 bytes in size. Each entry is initialized at a specific memory address in hexadecimal. Then, for example:

$$
\begin{aligned}
[0] &= 0000\ 0000\ 0000\ 0000 \quad \text{at memory address 0x4a0} \\
[1] &= 0000\ 0000\ 0000\ 0000 \quad \text{at memory address 0x4a4} \\
[2] &= 0000\ 0000\ 0000\ 0000 \quad \text{at memory address 0x4a8} \\
[3] &= 0000\ 0000\ 0000\ 0000 \quad \text{at memory address 0x4ac}
\end{aligned}
$$

The array starts at address 0x4a0. Since hexadecimal is base-16, each digit represents 4 bits. Thus, array cells are 4 bytes apart.

Function 5.2: Repeated-Squaring Algorithm

On input $\alpha \in \mathbb{Z}_n$ and a non-negative integer e, where $e = (b_{\ell-1} \cdots b_0)_2$ is the binary expansion of e, the repeated-squaring algorithm computes $\alpha^e \mod n$.

Input: $\alpha \in \mathbb{Z}_n$, $e \in \mathbb{Z}_{\geq 0}$ (binary)
Output: $\beta = \alpha^e \mod n$

```
1  Function RepeatedSquare(α, e):
2  |  β ← [1]_n;
3  |  for i ← ℓ − 1 down to 0 do
4  |  |  β ← β^2 mod n;
5  |  |  if b_i = 1 then
6  |  |  |  β ← β · α mod n;
7  |  |  end
8  |  end
9  |  return β;
10 return
```

Time Complexity: $O(\ell)$, where ℓ is the number of bits in e. This involves ℓ squarings and at most ℓ additional multiplications in $\mathbb{Z}_n$. **Space Complexity:** $O(\ell)$, where ℓ is the number of bits in e.

Example 3.5. Suppose $e = 37 = (100101)_2$. The above algorithm performs the following operations in this case:

$$\begin{aligned}
&\beta \leftarrow [1] && //0\\
&\beta \leftarrow \beta^2,\ \beta \leftarrow \beta \cdot \alpha && //1\\
&\beta \leftarrow \beta^2 && //10\\
&\beta \leftarrow \beta^2 && //100\\
&\beta \leftarrow \beta^2,\ \beta \leftarrow \beta \cdot \alpha && //1001\\
&\beta \leftarrow \beta^2 && //10010\\
&\beta \leftarrow \beta^2,\ \beta \leftarrow \beta \cdot \alpha && //100101
\end{aligned}$$

We utilize this algorithm to compute **multiplicative inverses** and test, **quadratic residues** and **primality**. The following algorithms demonstrate such; Though in the following chapters, we will discuss more optimal solutions.

Theorem 5.2: Computing Multiplicative Inverses in $\mathbb{Z}_p$

Let p be a prime and $\alpha \in \mathbb{Z}_p^*$. The multiplicative inverse of α, denoted α^{-1}, can be computed using Euler's theorem. We have:

$$\alpha^{p-1} = 1 \pmod{p}.$$

Multiplying both sides by α^{-1}, we obtain:

$$\alpha^{p-2} = \alpha^{-1} \pmod{p}.$$

Thus, α^{-1} is computed by raising α to the power $p-2$.

Theorem 5.3: Testing Quadratic Residuosity

Let p be an odd prime and $\alpha \in \mathbb{Z}_p^*$. To test whether $\alpha \in (\mathbb{Z}_p^*)^2$, we apply Euler's criterion:

$$\alpha \in (\mathbb{Z}_p^*)^2 \iff \alpha^{(p-1)/2} = 1 \pmod{p}.$$

Therefore, to determine whether α is a quadratic residue, we raise α to the power $(p-1)/2$.

Theorem 5.4: Computing Multiplicative Inverses in $\mathbb{Z}_p$

Let p be a prime and $\alpha \in \mathbb{Z}_p^*$. The multiplicative inverse of α, denoted α^{-1}, can be computed using Euler's theorem. We have:

$$\alpha^{p-1} = 1 \pmod{p}.$$

Multiplying both sides by α^{-1}, we obtain:

$$\alpha^{p-2} = \alpha^{-1} \pmod{p}.$$

Thus, α^{-1} is computed by raising α to the power $p-2$.

Our next algorithm will optimize multiplication between two large integers. Our current method of multiplying a, b and then dividing costs: $Mul(a, b) = O(k\ell)$ where $k = ||a||$ and $\ell = ||b||$. Then for, $k = \ell = n$, we have $O(n^2)$, taking quadratic time.

Theorem 5.5: Karatsuba's Multiplication Algorithm

Let P and Q be large integers of length $2k$. Then $P \cdot Q$ can be computed in $O(n^{\log_2 3})$:

$$P \cdot Q = P_1Q_1 \cdot 10^{2k} + [(P_1 + P_2)(Q_1 + Q_2) - P_1Q_1 - P_2Q_2] \cdot 10^k + P_2Q_2.$$

where subscripts 1 and 2 refer to the higher and lower orders, where P_1Q_1 are P_2Q_2 are computed once.

Proof 5.1: Karatsuba's Multiplication Algorithm

Step 1: Divide the numbers: Split the large integers P and Q into two halves:

$$P = P_1 \cdot 10^k + P_2 \quad \text{and} \quad Q = Q_1 \cdot 10^k + Q_2,$$

where P_1, P_2, Q_1, Q_2 represent the high and low parts of the numbers, and k is half the number of digits.

Step 2: Expand the product: The classical multiplication formula for $P \cdot Q$ is:

$$P \cdot Q = P_1Q_1 \cdot 10^{2k} + (P_1Q_2 + P_2Q_1) \cdot 10^k + P_2Q_2.$$

We compute: P_1Q_1 and P_2Q_2, holding off on $(P_1Q_2 + P_2Q_1)$.

Step 3: Karatsuba's Optimization: We reduce the number of multiplications by introducing the term: $(P_1 + P_2)(Q_1 + Q_2)$.
Expanding this, we have:

$$(P_1 + P_2)(Q_1 + Q_2) = P_1Q_1 + P_1Q_2 + P_2Q_1 + P_2Q_2.$$

Note, we already have P_1Q_1 and P_2Q_2. Let $P' := P_1Q_1$ and $Q' := P_2Q_2$. Then,

$$(P_1Q_2 + P_2Q_1) := ((P_1 + P_2)(Q_1 + Q_2)) - P' - Q'$$

Where $((P_1 + P_2)(Q_1 + Q_2))$ is **one multiplication**, reducing multiplications to 3.

Step 4: Combine the results. Using the above observations, we can rewrite the product of $P \cdot Q$ as:

$$P \cdot Q = P_1Q_1 \cdot 10^{2k} + [(P_1 + P_2)(Q_1 + Q_2) - P_1Q_1 - P_2Q_2] \cdot 10^k + P_2Q_2.$$

This results in only 3 recursive multiplications and achieves a time complexity of $O(n^{\log_2 3})$, which is approximately $O(n^{1.585})$, making it much faster than $O(n^2)$.

■

— 4 —

Computational Efficiency

4.1 Euclid's Algorithm

The following algorithms we will discuss seeks to compute the greatest common divisor of two integers. The first algorithm we will discuss is **Euclid's Algorithm**.

Theorem 1.1: Euclid's Strategy

Let $a, b \in \mathbb{Z}$ with $a \geq b \geq 0$. To compute $\gcd(a, b)$:

(i.) If $b = 0$, then $\gcd(a, b) = a$.

(ii.) If $b > 0$, then $q := \lfloor a/b \rfloor$ and $r := a \mod b$, from the Division Algorithm, $a = bq + r$.

a) If $d \mid b$ and $d \mid r$, then $d \mid a$.

b) If $d \mid a$ and $d \mid b$, then $d \mid r$.

c) Then $\gcd(a, b) = \gcd(b, r)$.

1. Compute the reduced form $\gcd(b, r)$, to yield $\gcd(a, b)$.

Before we proceed, we will introduce ϕ, not to be confused with φ (Euler's totient function)(4.1).

Definition 1.1: ϕ - The Golden Ratio

The symbol ϕ (the Greek letter "phi") refers to the **golden ratio**, defined as:

$$\phi := \frac{1 + \sqrt{5}}{2} \approx 1.618.$$

In relation to the Fibonacci sequence, as n increases, the ratio of consecutive Fibonacci numbers approaches ϕ:

$$\lim_{n \to \infty} \frac{F_n}{F_{n-1}} = \phi,$$

where F_n represents the n-th Fibonacci number. This shows that the Fibonacci sequence asymptotically grows by a factor of ϕ between successive terms.

Function 1.1: Euclid's Algorithm - *gcd()*

Let a and b be integers such that $a \geq b \geq 0$, compute $d = \gcd(a, b)$ as follows:

Input: a, b (integers with $a \geq b \geq 0$)
Output: $d = \gcd(a, b)$

```
1  Function gcd(a, b):
2      remainder ← a;
3      divisor ← b;
4      while divisor ≠ 0 do
5          newRemainder ← remainder mod divisor;
6          remainder ← divisor;
7          divisor ← newRemainder;
8      end
9      gcd ← remainder;
10     return gcd;
11 return
```

Time Complexity: $O(||a|| \cdot ||b||)$. Our most expensive call only even happens on the first iteration, where a and b are the largest. Let $n := ||a|| = ||b||$. Then one could assume each new remainder is one less than the previous, leading to $O(n)$ iterations, yielding $O(n^3)$. However, we will show that as we decrease $||b||$, iterations increase and costs decrease at a fixed ratio.

Proof 1.1: Euclidean Algorithm Time Complexity (Part 1)

Let $a, b \in \mathbb{Z}$ such that $a \geq b \geq 0$. We define the integers $r_0, r_1, \ldots, r_{\lambda+1}$ and $q_1, q_2, \ldots, q_\lambda$, as each new remainder and divisor each iterations, where $\lambda \geq 0$, as follows:

$$\begin{aligned}
a &= r_0, \\
b &= r_1, \\
r_0 &= r_1 q_1 + r_2 && (0 < r_2 < r_1), \\
r_1 &= r_2 q_2 + r_3 && (0 < r_3 < r_2), \\
&\vdots \\
r_{i-1} &= r_i q_i + r_{i+1} && (0 < r_{i+1} < r_i), \\
&\vdots \\
r_{\lambda-2} &= r_{\lambda-1} q_{\lambda-1} + r_\lambda && (0 < r_\lambda < r_{\lambda-1}), \\
r_{\lambda-1} &= r_\lambda q_\lambda + r_{\lambda+1} && (r_{\lambda+1} = 0).
\end{aligned}$$

By definition, $\lambda = 0$ if $b = 0$, and $\lambda > 0$ otherwise. Then, $r_\lambda = \gcd(a, b)$. Moreover, if $b > 0$, then $\lambda \leq \log b / \log \phi + 1$, where $\phi := \frac{1+\sqrt{5}}{2} \approx 1.62$ is the golden ratio.

■

Proof 1.2: Euclidean Algorithm Time Complexity (Part 2)

For the first statement, one sees that for $i = 1, \ldots, \lambda$, we have $r_{i-1} = r_i q_i + r_{i+1}$, from which it follows that the common divisors of r_{i-1} and r_i are the same as the common divisors of r_i and r_{i+1}, and hence $\gcd(r_{i-1}, r_i) = \gcd(r_i, r_{i+1})$. From this, it follows that

$$\gcd(a, b) = \gcd(r_0, r_1) = \cdots = \gcd(r_\lambda, r_{\lambda+1}) = \gcd(r_\lambda, 0) = r_\lambda.$$

To prove the second statement, assume that $b > 0$, and hence $\lambda > 0$. If $\lambda = 1$, the statement is obviously true, so assume $\lambda > 1$. We claim that for $i = 0, \ldots, \lambda - 1$, we have $r_{\lambda-i} \geq \phi^i$. The statement will then follow by setting $i = \lambda - 1$ and taking logarithms.

We now prove the above claim. For $i = 0$ and $i = 1$, we have

$$r_\lambda \geq 1 = \phi^0 \quad \text{and} \quad r_{\lambda-1} \geq r_\lambda + 1 \geq 2 \geq \phi^1.$$

For $i = 2, \ldots, \lambda - 1$, using induction and applying the fact that $\phi^2 = \phi + 1$, we have

$$r_{\lambda-i} \geq r_{\lambda-(i-1)} + r_{\lambda-(i-2)} \geq \phi^{i-1} + \phi^{i-2} = \phi^{i-2}(1 + \phi) = \phi^i,$$

which proves Part 1.

■

Example: Suppose $a = 100$ and $b = 35$. Then the numbers appearing in Theorem 4.1 are easily computed as follows:

i	0	1	2	3	4
r_i	100	35	30	5	0
q_i		2	1	6	

So we have $\gcd(a, b) = r_3 = 5$.

Proof 1.3: Euclidean Algorithm Time Complexity (Part 3)

We may assume that $b > 0$. With notation as in Theorem 4.1, the running time is $O(T)$, where

$$T = \sum_{i=1}^{\lambda} \text{len}(r_i)\text{len}(q_i) \leq \text{len}(b) \sum_{i=1}^{\lambda} \text{len}(q_i) \leq \text{len}(b) \sum_{i=1}^{\lambda} (\text{len}(r_{i-1}) - \text{len}(r_i) + 1)$$

$$= \text{len}(b)(\text{len}(r_0) - \text{len}(r_\lambda) + \lambda) \quad \text{(telescoping the sum)}$$

$$\leq \text{len}(b)(\text{len}(a) + \log b / \log \phi + 1) \quad \text{(by Theorem 4.1)}$$

$$= O(\text{len}(a)\,\text{len}(b)).$$

■

4.2 Extended Euclidean Algorithm

Let $a, b \in \mathbb{Z}$ and $d := \gcd(a, b)$. Then by B'ezout's Identity (5.2), there exist $s, t \in \mathbb{Z}$ such that $as + bt = d$. For which the **Extended Euclidean Algorithm** computes s and t.

Theorem 2.1: Euclid's Extended Algorithm Strategy

Let a, b, $r_0, \ldots, r_{\lambda+1}$ and $q_1, \ldots, q_\lambda$ be as in Proof (1.1). Define integers $s_0, \ldots, s_{\lambda+1}$ and $t_0, \ldots, t_{\lambda+1}$ as follows:

$$\begin{aligned} s_0 &:= 1, & t_0 &:= 0, \\ s_1 &:= 0, & t_1 &:= 1, \\ s_{i+1} &:= s_{i-1} - s_i q_i, & t_{i+1} &:= t_{i-1} - t_i q_i, \quad (i = 1, \ldots, \lambda). \end{aligned}$$

Then:

(i) for $i = 0, \ldots, \lambda + 1$, we have $as_i + bt_i = r_i$; in particular, $as_\lambda + bt_\lambda = \gcd(a, b)$;

(ii) for $i = 0, \ldots, \lambda$, we have $s_i t_{i+1} - t_i s_{i+1} = (-1)^i$;

(iii) for $i = 0, \ldots, \lambda + 1$, we have $\gcd(s_i, t_i) = 1$;

(iv) for $i = 0, \ldots, \lambda$, we have $t_i t_{i+1} \leq 0$ and $|t_i| \leq |t_{i+1}|$; for $i = 1, \ldots, \lambda$, we have $s_i s_{i+1} \leq 0$ and $|s_i| \leq |s_{i+1}|$;

(v) for $i = 1, \ldots, \lambda + 1$, we have $r_{i-1}|t_i| \leq a$ and $r_{i-1}|s_i| \leq b$;

(vi) if $a > 0$, then for $i = 1, \ldots, \lambda + 1$, we have $|t_i| \leq a$ and $|s_i| \leq b$; if $a > 1$ and $b > 0$, then $|t_\lambda| \leq a/2$ and $|s_\lambda| \leq b/2$.

Proof 2.1: Extended Euclidean Algorithm (Part 1)

(i) By induction on i, with $i = 0, 1$. For $i = 2, \ldots, \lambda + 1$, we have

$$\begin{aligned} as_i + bt_i &= a(s_{i-2} - s_{i-1}q_{i-1}) + b(t_{i-2} - t_{i-1}q_{i-1}) \\ &= (as_{i-2} + bt_{i-2}) - (as_{i-1} + bt_{i-1})q_{i-1} \\ &= r_{i-2} - r_{i-1}q_{i-1} = r_i \quad \text{(by induction)} \end{aligned}$$

(ii) By induction on i, with $i = 0$. Then $i = 1, \ldots, \lambda$, we have

$$\begin{aligned} s_i t_{i+1} - t_i s_{i+1} &= s_i(t_{i-1} - t_i q_i) - t_i(s_{i-1} - s_i q_i) \\ &= -(s_i t_i - t_i s_i) \\ &= -(1) = (-1)^i \quad \text{(by induction).} \end{aligned}$$

■

Proof 2.2: Extended Euclidean Algorithm (Part 2)

(iii) follows directly from (ii) from Part 1.
(iv) By induction on i. The statement involving the t_i's is true for $i = 0$. For $i = 1, \ldots, \lambda$, we have

$$t_{i+1} = t_i - t_i q_i;$$

moreover, by the induction hypothesis, t_{i-1} and t_i have opposite signs and $|t_i| \geq |t_{i-1}|$. It follows that $|t_{i+1}| = |t_{i-1}| + |t_i|q_i \geq |t_i|$, and that the sign of t_{i+1} is the opposite of that of t_i. The proof of the statement involving the s_i's is the same, except that we start the induction at $i = 1$.
For (v), one considers the two equations:

$$as_{i-1} + bt_{i-1} = r_{i-1},$$

$$as_i + bt_i = r_i.$$

Subtracting t_{i-1} times the second equation from t_i times the first, and applying (ii), we get

$$\pm a = t_i r_{i-1} - t_{i-1} r_i;$$

consequently, using the fact that t_i and t_{i-1} have opposite signs, we obtain

$$a = |t_i r_{i-1} - t_{i-1} r_i| = |t_i||r_{i-1}| + |t_{i-1}||r_i| \geq |t_i||r_{i-1}|.$$

The inequality involving s_i follows similarly, subtracting s_{i-1} times the second equation from s_i times the first.
(vi) follows from (v) and the following observations: if $a > 0$, then $r_{i-1} > 0$ for $i = 1, \ldots, \lambda+1$; and if a and $b > 0$, then $\lambda > 0$ and $r_{\lambda-1} \geq 2$.

■

Example: We continue with the previous example. The s_i's and t_i's are computed from the q_i's:

i	r_i	q_i	s_i	t_i
0	100		1	0
1	35	2	0	1
2	30	1	1	-2
3	5	6	-1	3
4	0		7	-20

So we have $\gcd(a, b) = 5 = -a + 3b$.

Function 2.1: Extended Euclidean Algorithm - *gcdExtended()*

On input a, b, where a and b are integers such that $a \geq b \geq 0$, compute integers d, s, and t, such that $d = \gcd(a, b)$ and $as + bt = d$, as follows:

Input: a, b (integers with $a \geq b \geq 0$)
Output: $d = \gcd(a, b)$, and integers s, t such that $as + bt = d$

Function *gcdExtended(a, b)*:
  $r \leftarrow a$, $r' \leftarrow b$;
  $s \leftarrow 1$, $s' \leftarrow 0$;
  $t \leftarrow 0$, $t' \leftarrow 1$;
  while $r' \neq 0$ **do**
    $q \leftarrow \left\lfloor \frac{r}{r'} \right\rfloor$;
    $r'' \leftarrow r \mod r'$;
    $(r, r', s, s', t, t') \leftarrow (r', r'', s', s - s'q, t', t - t'q)$;
  end
  $d \leftarrow r$;
  return d, s, t;
return

Time Complexity: $O(\|a\| \cdot \|b\|)$, where $||a||$ and $||b||$ are the number of bits in a and b. We may assume that $b > 0$. It suffices to analyze the cost of computing the coefficient sequences $\{s_i\}$ and $\{t_i\}$. Consider first the cost of computing all of the t_i's, which is $O(T)$, where $T = \sum_{i=2}^{\lambda} \|t_i\| \cdot \|q_i\|$. We have $t_1 = 1$ and, by part (vi) of our previous proof, we have $|t_i| \leq a$ for $i = 2, \ldots, \lambda$. Then we have

$$T \leq \|q_1\| + \|a\| \sum_{i=2}^{\lambda} \|q_i\|$$

$$\leq \|a\| + \|a\|(\|r_1\| - \|r_\lambda\| + \lambda - 1) = O(\|a\| \cdot \|b\|).$$

An analogous argument shows that one can also compute all of the s_i's in time $O(\|a\| \cdot \|b\|)$, and in fact, in time $O(\|b\|^2)$.