My cryptography book

John Doe (March 17, 2023)

Contents

Chapter 1

Basic number theory

SUGGESTIONS. For this chapter, state the basic axioms and properties/theorems of $\mathbb{Z}$. Provide proofs. But remember that most of the properties/theorems can be generlized to properties/theorems for rings. It's still a good idea to prove the facts for $\mathbb{Z}$ since $\mathbb{Z}$ is not as abstract as general rings and will prepare you for the general results.

1.1 Axioms of $\mathbb{Z}$

We will assume that $(\mathbb{Z}, +, \cdot, 0, 1)$ satisfies the following axioms.

- PROPERTIES OF $+$:
  - Closure: If $x, y \in \mathbb{Z}$, then $x + y \in \mathbb{Z}$.
  - Associativity: If $x, y, z \in \mathbb{Z}$, then $(x + y) + z = x + (y + z)$.
  - Inverse: If $x \in \mathbb{Z}$, then there is some y such that $x + y = 0 = y + x$. The y in the above is an additive inverse of x.
  - Neutrality: If $x \in \mathbb{Z}$, then $0 + x = x = x + 0$.
  - Commutativity: If $x, y \in \mathbb{Z}$, then $x + y = y + x$.

  (Memory aid for the first four: CAIN.)
- PROPERTIES OF $\cdot$:
  - Closure: If $x, y \in \mathbb{Z}$, then $x \cdot y \in \mathbb{Z}$.
  - Associativity: If $x, y, z \in \mathbb{Z}$, then $(x \cdot y) \cdot z = x \cdot (y \cdot z)$.
  - Neutrality: If $x \in \mathbb{Z}$, then $1 \cdot x = x = x + 1$.
  - Commutativity: If $x, y \in \mathbb{Z}$, then $x \cdot y = y \cdot x$.

  It is common to write xy instead of $x \cdot y$.
- DISTRIBUTIVITY: If $x, y, z \in \mathbb{Z}$, then $x \cdot (y + z) = x \cdot y + x \cdot z$ and $(y + z) \cdot x = y \cdot x + z \cdot x$

A set R with operations $+, \cdot$ and elements $0_R, 1_R$ satisfying the above properties

the above is called a commutative ring. This is an important generalization because there are many very useful commutative rings and we want to prove results about commutative rings so that these results can be applied to all commutative rings, including but not restricted to $\mathbb{Z}$. And if the commutativity of multiplication is left out, then we have the concept of a ring; for emphasize these are called non-commutative rings. This is also an important concept since $n \times n$ matrices with $\mathbb{R}$ entries, $\mathrm{M}_{n\times n}(\mathbb{R})$, form a non-commutative ring. In fact, more generally, the set of $n \times n$ matrices with entries in a commutative ring R, $\mathrm{M}_{n\times n}(R)$, is itself a non-commutative ring. We will return to the concept of commutative and non-commutative rings later.

The next property of $\mathbb{Z}$, integrality, is very special and does not apply to many communtative rings and is therefore left out of the definition of commutative ring:

- Integrality: If $x, y \in \mathbb{Z}$, then $xy = 0 \implies x = 0$ or $y = 0$.

Another property of $\mathbb{Z}$ that we will assume is

- Nontriviality: $0 \neq 1$

This axiom of $\mathbb{Z}$ is extremely simple, but cannot be deduced from the previous axioms.

The above forms the algebraic properties of $\mathbb{Z}$, i.e., properties involving addition and multiplication.

It is actually possible to first define axioms for $\mathbb{N} = \{0, 1, 2, ...\}$ and then define $\mathbb{Z}$ in terms of $\mathbb{N}$. We will not do that except to mention that the axioms for $\mathbb{N}$ are called the Peano-Dedekind axioms and that one very important Peano-Dedekind axiom of $\mathbb{N}$ is the

- Well-ordering principle (WOP) for $\mathbb{N}$: If X is a nonempty subset of $\mathbb{N}$, then X contains a minimum element, i.e., there is some $m \in X$ such that
$$m \leq x$$
for all $x \in X$.

Without going into details, it can be shown that for $\mathbb{N}$, the WOP is equivalent to each of the following axioms:

- Weak Mathematical Induction for $\mathbb{N}$: Let X be a subset of $\mathbb{N}$ satisyfing the following two conditions:

  - $0 \in X$ and
  - Let $n \in \mathbb{N}$. If $n \in X$, then $n+1 \in X$.

  Then $X = \mathbb{N}$.

and

- Strong Mathematical Induction for $\mathbb{N}$: Let X be a subset of $\mathbb{N}$ satisyfing the following two conditions:
  - $0 \in X$ and
  - Let $n \in \mathbb{N}$. If $k \in X$ for all $0 \leq k \leq n$, then $n+1 \in X$.

  Then $X = \mathbb{N}$.

In the above two induction axioms, if we write $X = \{n \mid P(n)\}$ where $P(n)$ is a propositional formula, then the induction axioms can be rewritten in the following way:

- Weak Mathematical Induction: Let $P(n)$ be a proposition for $n \in \mathbb{N}$ satisyfing the following two conditions:
  - $P(0)$ is true and
  - Let $n \in \mathbb{N}$. If $P(n)$ is true, then $P(n+1)$ is true.

  Then $P(n)$ is true for all $n \in \mathbb{N}$.

and

- Strong Mathematical Induction: Let $P(n)$ be a proposition for $n \in \mathbb{N}$ satisyfing the following two conditions:
  - $P(0)$ is true and
  - Let $n \in \mathbb{N}$. If $k \in X$ for all $0 \leq k \leq n$, then $n+1 \in X$.
  - Let $n \in \mathbb{N}$. If $P(k)$ is true for $0 \leq k \leq n$, then $P(n+1)$ is true.

  Then $P(n)$ is true for all $n \in \mathbb{N}$.

The above are the algebraic axioms of $\mathbb{Z}$. There's also the order relation of $\mathbb{Z}$ which is used in WOP and the two induction principles. I will formalize the axioms of the order relation later. For now one can assume that the order relation is defined as follows: If $x \in \mathbb{Z}$, then

$$x < y$$

if there is some $z \in \mathbb{N}$ such that

$$x + z = y$$

There is one more axiom of $\mathbb{Z}$ that is related to the "topology" of $\mathbb{Z}$ and uses the order relation:

- Topology: Given any $x \in \mathbb{Z}$, there is no $y \in \mathbb{Z}$ such that

$$x < y < x + 1$$

You can think of topology of a set as study of "closeness" of values in that set. For $\mathbb{Q}$, given any two distinct rational values $x < y$, there is also some $z \in \mathbb{Q}$ such that $x < z < y$. This is the same for $\mathbb{R}$. Therefore the topology of $\mathbb{Z}$ is very different from the topology of $\mathbb{Q}$ and $\mathbb{R}$ because there are "holes" in $\mathbb{Z}$ where there are no $\mathbb{Z}$ values. $\mathbb{Z}$ has what is called a discrete topology.

The above assume the existence of an order relation on $\mathbb{Z}$, i.e., $<$. We have to include the following axioms of $<$ on $\mathbb{Z}$. There is a set $\mathbb{Z}^+$ such that the following holds:

- Trichotomy: If $x \in \mathbb{Z}$, then exactly one of the following holds: $-x \in \mathbb{Z}^+$, $x = 0$, $x \in \mathbb{Z}^+$.
- Closure of $+$: If $x, y \in \mathbb{Z}^+$, then $x + y \in \mathbb{Z}^+$.
- Closure of $\cdot$: If $x, y \in \mathbb{Z}^+$, then $x \cdot y \in \mathbb{Z}^+$.

We then define $<$ as follows: If $x, y \in \mathbb{Z}$, then we write $x < y$ if

$$y - x \in \mathbb{Z}^+$$

Since $<$ is defined, we can define $x \leq y$ to mean "either $x < y$ or $x = y$". The above order relation is expressed abstractly without refering to the fact that $\mathbb{Z}^+$ is $\{1, 2, 3, ...\}$, i.e., the set of positive integers. In fact, you can prove $\mathbb{Z}^+ = \{1, 2, 3, ...\}$ from the above axioms – see exercises below.

Proposition 1.1.1. *The additive inverse for x is unique. In other words, if y, y' satisfies*

$$x + y = 0 = y + x$$
$$x + y' = 0 = y' + x$$

Then $y = y'$.

Proof. TODO

Since the additive inverse of x is unique, we can choose to write the additive inverse of x in terms of x. This is usually written $-x$. We define the operator $-$ in terms of the additive inverse:

Definition 1.1.1. Let $x, y \in \mathbb{Z}$. We define the subtraction operator as

$$x - y = x + (-y)$$

Note that every x in $\mathbb{Z}$ as an additive inverse, but we did not require value of $\mathbb{Z}$ to have multiplicative inverse:

Definition 1.1.2. Let $x \in \mathbb{Z}$. Then y is a multiplicative inverse of x if

$$x \cdot y = 1 = y \cdot x$$

We say that x is a **unit** if x has a multiplicative inverse. We can also say that x is **invertible**.

unit

invertible

Intuitively, you know that the only values of $\mathbb{Z}$ with multiplicative inverses are 1 and -1.

Proposition 1.1.2. *Let $x \in \mathbb{Z}$. If x is a unit, then the multiplicative inverse of x is unique. In other words if y, y' satisfies*

$$xy = 1 = yx$$
$$xy' = 1 = y'x$$

then $y = y'$.

Definition 1.1.3. The multiplicative inverse of x, it is exists, is is denoted by x^{-1}.

Proposition 1.1.3. *Cancellation law for addition. Let $x, y, z \in \mathbb{Z}$.*

(a) *If $x + z = y + z$, then $x = y$.*
(b) *If $z + x = z + y$, then $x = y$.*

Proposition 1.1.4. *Let $x \in \mathbb{Z}$.*

(a) $0x = 0 = x0$

(b) $-0 = 0$
(c) $x - 0 = x$. *(NOTE CORRECTION)*

Proof. (a) We will first prove $0x = 0$:

$$\begin{aligned} 0x &= (0+0)x && \text{by Neutrality of } + \\ &= 0x + 0x && \text{by Distributivity} \end{aligned} \tag{1}$$

Since $0x \in \mathbb{Z}$ by Closure of $\cdot$, there is exists some $y \in \mathbb{Z}$ which is an additive inverse of $0x$, i.e.,

$$0x + y = 0 = y + 0x \tag{2}$$

From (1),

$$\begin{aligned} y + 0x &= y + (0x + 0x) \\ 0 &= y + (0x + 0x) && \text{by (2)} \\ 0 &= (y + 0x) + 0x && \text{by Associativity of } + \\ 0 &= 0 + 0x && \text{by (2)} \\ 0 &= 0x && \text{by Neutrality of } + \end{aligned}$$

To prove $0 = x0$, from above

$$\begin{aligned} 0 &= 0x \\ &= x0 && \text{by Commutativity of } \cdot \end{aligned}$$

(b) TODO

(c) TODO □

Proposition 1.1.5. *Let $x, y \in \mathbb{Z}$.*

(a) $-(-1) = 1$
(b) $-(-x) = x$
(c) $x(-1) = -x = (-1)x$
(d) $(-1)(-1) = 1$
(e) $(-x)(-y) = xy$

Proposition 1.1.6. *Cancellation law for multiplication. Let $x, y, z \in \mathbb{Z}$.*

(a) *If $xz = yz$ and $z \neq 0$, then $x = y$.*

(b) *If* $zx = zy$ *and* $z \neq 0$*, then* $x = y$*.*

For convenience, I will write $x^2 = xx$ and in general

$$x^n = \begin{cases} 1 & \text{if } n = 0 \\ x^{n-1}x & \text{if } n > 0 \end{cases}$$

If x has a multiplicative inverse, i.e., if x^{-1} exists, then, for $n \geq 0$, I will define

$$x^{-n} = \left(x^{-1}\right)^n$$

Proposition 1.1.7. *Let* $x \in \mathbb{Z}$*. Then* $[n]x = n \cdot x$*.*

Note that nx has two meanings: nx can be the multiplication of n and x and it can also be $x + \cdots + x$ with n number of x. Of course you would expect them to be the same. For now define

$$[n]x = \begin{cases} 0 & \text{if } n = 0 \\ [n-1]x + x & \text{if } n > 0 \end{cases}$$

and if n is negative, we define

$$[n]x = -([-n]x)$$

1.2 Divisibility

Definition 1.2.1. Let $m, n \in \mathbb{Z}$. Then we say that m **divides** n, and we write $m \mid n$, if there is some $x \in \mathbb{Z}$ such that $mx = n$, i.e.,

$$\exists x \in \mathbb{Z} \cdot [mx = n]$$

Proposition 1.2.1. *Let $a \in \mathbb{Z}$.*

(a) *Then $1 \mid a$.*
(b) *Then $a \mid 0$.*
(c) *(Reflexive) $a \mid a$.*
(d) *If $a \mid b$ and $b \mid a$, then $a = \pm b$.*
(e) *(Transitive) If $a \mid b$ and $b \mid c$, then $a \mid c$.*
(f) *If $a \mid b$, then $a \mid bc$.*
(g) *If $a \mid b$, $a \mid c$, then $a \mid b + c$.*
(h) *(Linearity) If $a \mid b$, $a \mid c$, then $a \mid bx + cy$ for $x, y \in \mathbb{Z}$.*

Proof. TODO

1.3 Congruences

Definition 1.3.1. Let $a, b \in \mathbb{Z}$ and $N \in \mathbb{Z}$ with $N > 0$. Then a is congruent to b mod N and we write

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

if $N \mid a - b$.

Proposition 1.3.1. *Let $a, b, c, a', b' \in \mathbb{Z}$.*

(a) *(Reflexivity)* $a \equiv a \pmod{N}$
(b) *(Symmetry) If* $a \equiv b \pmod{N}$, *then* $b \equiv a \pmod{N}$
(c) *(Transitivity) If* $a \equiv b, b \equiv c \pmod{N}$, *then* $a \equiv c \pmod{N}$
(d) *If* $a \equiv b$, $a' \equiv b' \pmod{N}$, *then* $a + a' \equiv b + b' \pmod{N}$.
(e) *If* $a \equiv b$, $a' \equiv b' \pmod{N}$, *then* $aa' \equiv bb' \pmod{N}$.

Proof. TODO

Proposition 1.3.2. *Let $a, N \in \mathbb{Z}$ with $N > 0$. Let $q, r \in \mathbb{Z}$ such that*

$$a = Nq + r, \quad 0 \leq r < N$$

Then $a \equiv r \pmod{N}$.

Proof. TODO

Exercise 1.3.1. Show that the cancellation law for $\mathbb{Z}$ does not translate to $\mathbb{Z}$ mod N. In other words, find N, a, b, c such that $c \not\equiv 0 \pmod{N}$ and

$$ac \equiv bc \pmod{N}, \quad a \not\equiv b \pmod{N}$$

1.4 Euclidean property

$\mathbb{Z}$ satisfies this very important property:

Euclidean property

Theorem 1.4.1. (**Euclidean property**) *If $a, b \in \mathbb{Z}$ with $b \neq 0$, then there are integers q and r satisfying*

$$a = bq + r, \quad 0 \leq |r| < |b|$$

The above theorem is the version that can be generalized to general rings. Below is the version for $\mathbb{Z}$. The only difference is the $|r|$ is replaced by r:

Euclidean property 2

Theorem 1.4.2. (**Euclidean property 2**) *If $a, b \in \mathbb{Z}$ with $b \neq 0$, then there are integers q and r satisfying*

$$a = bq + r, \quad 0 \leq r < |b|$$

In many cases, one is interested in the case when $a \geq 0$. So this version is the one found in most textbooks:

Euclidean property 3

Theorem 1.4.3. (**Euclidean property 3**) *If $a, b \in \mathbb{Z}$ with $a \geq 0, b > 0$, then there are integers $q \geq 0$ and $r \geq 0$ satisfying*

$$a = bq + r, \quad 0 \leq r < b$$

quotient
remainder

q is called the **quotient** when a is divided by b; r is the **remainder**. q and r are unique (see proposition below). For instance if $a = 25$ and $b = 3$, then

$$25 = 3 \cdot 8 + 1, \quad 0 \leq 1 < 3$$

The computation

$$a, b \to q, r$$

division algorithm

is called a **division algorithm**.

In Python, you can do this:

```
a = 25
b = 8
q, r = divmod(25, 8)
print("%s = %s * %s + %s" % (a, b, q, r))
```

```
[student@localhost ciss451-book-project] python divmod.py
25 = 8 * 3 + 1
```

Algorithmically, when a and b have a huge number of digits and they are represented using arrays of digits, the division algorithm to compute q, r is basically long division you learnt in middle school. At the hardware level, the same division algorithm occurs but the computation is in terms of bits and not digits.

If we peek ahead and pretend for the time being that fractions such as $\frac{a}{b}$ exists, then for $a > 0$ and $b > 0$, we have

$$q = \left\lfloor \frac{a}{b} \right\rfloor, \quad r = a - bq$$

where $\lfloor x \rfloor$ means the floor of x. If we write `(a/b)` for the *integer* quotient of a by b (i.e. this is the / in C++ for integers) and `(a%b)` for the corresponding remainder, then of course we have

```
a = b * (a/b) + (a%b)
```

Although the above Euclidean property is for $\mathbb{Z}$, We will first prove it for $a \geq 0$ and $b > 0$. The q, r will satisfy $q \geq 0$, $r \geq 0$. (Furthermore in this setup q, r are unique.) Once we have proven the Euclidean property for integer $a \geq 0$, it will not be difficult to extend the result to the whole of $\mathbb{Z}$.

To prove the Euclidean property of $\mathbb{Z}$, we will use WOP. (One can also prove the Euclidean property of $\mathbb{Z}$ using induction.)

Well-ordering principle for $\mathbb{N}$

Well-ordering principle for $\mathbb{N}$: Let X be a nonempty subset of $\mathbb{N}$. Then X has a minimal element. In other words there is some $m \in X$ such that $m \leq x$ for all $x \in X$.

You can prove the following version of well-ordering principle on $\mathbb{Z}$:

Well-ordering principle for $\mathbb{Z}$

Well-ordering principle for $\mathbb{Z}$: Let X be a nonempty subset of $\mathbb{Z}$ that is *bounded below*. Then X has a minimal element. In other words there is

some $m \in X$ such that $m \leq x$ for all $x \in X$.

$\mathbb{R}$ does not satisfy the second version well-ordering principle with $\mathbb{Z}$ replaced by $\mathbb{R}$. For instance the open interval $X = (0, 1)$ is bounded below (for instance by -42). However there is no m in X such that $m \leq x$ for all x in X. For instance $m = 0.01 \in X$ is not a minimum element of X since $0.0001 \in X$ is smaller than m. Also, $m = 0.0000001 \in X$ is also not a minimum of X since $0.0000000001 \in X$ is less than m. In fact for any $m \in X$, $(1/2)m$ is in X and is less than m. In other words no value in X can be a minimum value of X.

Now we will prove Theorem 1.4.3.

Proof. TODO

Proposition 1.4.1. *Given a, b, the q, r in Theorem 1.4.3 are unique. In other words, if*

$$\begin{aligned} a &= bq + r, \;\; 0 \leq r < |b| \\ a &= bq' + r', \;\; 0 \leq r' < |b| \end{aligned}$$

then

$$q = q', \quad r = r'$$

Proof. From $bq + r = a = bq' + r'$, we have

$$bq + r = bq' + r'$$

If $q = q'$, then $r = r'$. We now assume $q \neq q'$. Without loss of generality, we'll assume that $q > q'$. We have

$$r' = b(q - q') + r > b + r \geq b$$

which contradicts $r' < b$. □

Now I'm going to prove Theorem 1.4.1 which allows a to be any integer.

Proof of Theorem 1.4.1. Now I'll use Euclidean Property 3 to prove Euclidean Property 1. We just need to handle the case when $a < 0$. Let u be ± 1 so that $ua \geq 0$. Let v be ± 1 so that $vb > 0$. Note that $(\pm 1)^2 = 1$, i.e., $u^{-1} = u, v^{-1} = v$. Using Euclidean Property 3, there exist $q' \geq 0, r'$ such that

$$a' = b'q' + r', \;\; 0 \leq r' < b'$$

Then

$$ua = vbq' + r', \;\; 0 \leq r' < vb = |b|$$

Multiplying by u^{-1}

$$a = uvbq' + ur', \;\; 0 \leq r' < vb = |b|$$

and hence

$$a = b(uvq') + ur', \;\; 0 \leq |ur'| < vb = |b|$$

(Note that $r' \geq 0$ and hence $|ur'| = |u||r'| = r'$.) Hence if $q = uvq'$ and $r = ur'$, then

$$a = bq + r, \;\; 0 \leq |r| < |b|$$

and we are done. □

Exercise 1.4.1. Using the Euclidean property, prove that every integer is congruence to $0, 1, 2,$ or $3 \mod 4$.

Exercise 1.4.2. Prove that squares are 0 or 1 mod 4. In other words if $a \in \mathbb{Z}$, then $a^2 \equiv 0$ or $1 \pmod 4$.

Exercise 1.4.3. Solve $4x^3 + y^2 = 5z^2 + 6$ (in $\mathbb{Z}$).

Exercise 1.4.4. Prove that $11, 111, 1111, 11111, 111111, ...$ are all not perfect squares. (An integer is a perfect square is it's of the form a^2 where a is an integer.)

Exercise 1.4.5. How many of $3, 23, 123, 1123, 11123, 111123, 1111123, ...$ are perfect squares?

1.5 Euclidean algorithm – GCD

Now let me use the Euclidean property to compute the gcd of two integers.

Let's use the division algorithm on 20 and 6.

$$20 = 6 \cdot 3 + 2, \quad 0 \leq 2 < 6$$

Suppose I want to compute $\gcd(20, 6)$. Of course the example is small enough that we know that it is 2. But notice something about this:

$$20 = 6 \cdot 3 + 2, \quad 0 \leq 2 < 6$$

If d is a divisor of 20 and 6, then it must also divide 2. Therefore $\gcd(20, 6)$ must divide 2. The converse might not be true. In general, we have this crucial bridge between Euclidean property and common divisors:

GCD Lemma

Lemma 1.5.1. (**GCD Lemma**) *If $a, b, q, r \in \mathbb{Z}$ such that*

$$a = bq + r$$

then

$$\{d \mid d \text{ is a common divisor of } a, b\} = \{d \mid d \text{ is a common divisor of } b, r\}$$

Hence

$$\gcd(a, b) = \gcd(b, r)$$

Proof. TODO

In particular, given $a, b \in \mathbb{Z}$ where $a > b > 0$. By the Euclidean property of $\mathbb{Z}$, there exist $q, r \in \mathbb{Z}$ such that

$$a = bq + r, \;\; 0 \leq r < b$$

Hence

$$\gcd(a, b) = \gcd(b, r)$$

Note that in the above, I only require $a = bq + r$. For instance for to $\gcd(120, 15)$, I can use $120 = 1 \cdot 15 + (120 - 15)$, i.e., $a = 120, b = 15, q = 1, r = 120 - 15$. Then $\gcd(120, 15) = \gcd(15, 120 - 15) = \gcd(15, 105)$.

However if I use the division algorithm, then r is "small":

$$0 \leq r < b$$

So if you want to compute $\gcd(a, b)$, make sure $a \geq b$ (otherwise swap them). Then $\gcd(a, b) = \gcd(b, r)$ and you would have $a \geq b > r$. So instead of computing $\gcd(a, b)$, you are better off computing $\gcd(b, r)$.

But like I said, we do not need the q and r to be the quotient and remainder. For instance suppose I want to compute the GCD of 514 and 24.

$$514 = 24 \cdot 1 + (514 - 24)$$

Then

$$\gcd(514, 24) = \gcd(24, 514 - 24)$$

which gives us

$$\gcd(514, 24) = \gcd(24, 490)$$

Note that $\gcd(0, n) = n$ for any positive integer n. I'll let you think about that one. (Remember what I said before: 0 is in some sense a big number, like a black hole. Because every positive number divides 0.)

Of course this gives rise to the following algorithm

```
ALGORITHM: GCD
Inputs: a, b
Output: gcd(a, b)

if b > a:
    swap a, b

if b == 0:
    return a
else:
    return GCD(a - b, b)
```

This only subtracts one copy of b from a. Suppose we can compute

$$a = bq + r, \quad 0 \leq r < b$$

Then

$$\gcd(a, b) = \gcd(b, r)$$

Of course r is the remainder when a is divided by b. Using this we rewrite the above code to get the **Euclidean Algorithm**:

Euclidean Algorithm

```
ALGORITHM: GCD (Euclidean algorithm)
Inputs: a, b
Output: gcd(a, b)

if b > a:
    # To make sure that for gcd(a,b), a >= b
    swap a, b

if b == 0:
    return a
else:
    return GCD(b, a % b)
```

Note that if `a` $<$ `b`, then

$$\texttt{GCD(a, b) = GCD(b, a \% b) = GCD(b, a)}$$

Therefore the swap is not necessary:

```
ALGORITHM: GCD (Euclidean algorithm)
Inputs: a, b
Output: gcd(a, b)

if b == 0:
    return a
else:
    return GCD(b, a % b)
```

In this case, I'm assuming that `a % b` is available. As an example:

$$\begin{aligned}\gcd(514, 24) &= \gcd(24, 514\%24) = \gcd(24, 10)\\ &= \gcd(10, 24\%10) = \gcd(10, 4)\\ &= \gcd(10, 10\%4) = \gcd(10, 2)\\ &= \gcd(2, 10\%2) = \gcd(2, 0)\\ &= 2\end{aligned}$$

The above can also be done in a loop:

```
ALGORITHM: GCD (Euclidean algorithm)
Inputs: a, b
Output: gcd(a, b)

while 1:
    if b == 0:
```

```
        return a
    else:
        a, b = b, a % b
```

Exercise 1.5.1. Compute the following using the Euclidean Algorithm explicitly.

(a) $\gcd(10, 1)$
(b) $\gcd(10, 10)$
(c) $\gcd(107, 5)$
(d) $\gcd(107, 26)$
(e) $\gcd(84, 333)$

Exercise 1.5.2. Compute the following. You should go as far as you can. In other words, either you can a fixed integer (such as 1) or derive the $\gcd(\alpha, \beta)$ where α, β are as simple as possible. For instance, to simplify $\gcd(3 + 2a, a)$, since $3 + 2a = 2 \cdot a + 3$, we have

$$\gcd(3 + 2a, a) = \gcd(a, 3)$$

In the following $a, b, x, n \in \mathbb{Z}$ are positive integers.

(a) $\gcd(ab, b)$
(b) $\gcd(a, a + 1)$
(c) $\gcd(ab + a, b)$ where $0 < a < b$
(d) $\gcd(a(a + 1) + a, (a + 1))$ where $0 < a < b$
(e) $\gcd(1 + x + \cdots + x^n, x)$
(f) $\gcd(F_{10}, F_{11})$ where F_n is the n–th Fibonacci number. (Recall: $F_0 = 1, F_1 = 1, F_{n+2} = F_{n+1} + F_n$ for $n \geq 0$.)

Despite the fact that the Euclidean algorithm is one of the fastest algorithm to compute the GCD of two numbers and has been discovered by Euclid a long time ago (BC 300), the actual runtime was not known until Lamé proved in 1844 that the number of steps to compute $\gcd(a, b)$ using the Euclidean algorithm is ≤ 5 times the number of digits (in base 10 notation) of $\min(a, b)$. For instance for the example above of $\gcd(514, 24)$, the number of digits of $\min(514, 24)$ is 2. Lamé theorem says that the number of steps made by the Euclidean algorithm in the computation of $\gcd(514, 24)$ is at most $5 \times 2 = 10$.

The actual number of steps in the earlier computation

$$\begin{aligned}\gcd(514, 24) &= \gcd(24, 514\%24) = \gcd(24, 10)\\ &= \gcd(10, 24\%10) = \gcd(10, 4)\\ &= \gcd(10, 10\%4) = \gcd(10, 2)\\ &= \gcd(2, 10\%2) = \gcd(2, 0)\\ &= 2\end{aligned}$$

is 4 (not counting the base case step), i.e.,

$$\gcd(514, 24) = \gcd(24, 10) = \gcd(10, 4) = \gcd(10, 2) = \gcd(2, 0)$$

Lamé's work is generally considered the beginning of computational complexity theory, which is the study of resources needed (time or space) to execute an algorithm. Another fascinating fact about Lamé's theorem is that historically the above proof is the first ever "use" of the Fibonacci sequence.

Theorem 1.5.1. (Lamé 1844) *Let $a > b > 0$ be integers. If the GCD computation of a, b using Euclidean algorithm results in n steps:*

$$\gcd(a_{n+1}, b_{n+1}) = \gcd(a_n, b_n) = \cdots = \gcd(a_1, b_1), \quad b_1 = 0$$

where $(a_{n+1}, b_{n+1}) = (a, b)$, and $a_i > b_i$, then

(a) *$a \geq F_{n+2}$ and $b \geq F_{n+1}$, where F_n are the Fibonacci numbers ($F_1 = 1, F_2 = 1, F_3 = 2, F_4 = 3, F_5 = 5$, etc. Note that the index starts with 1.)*
(b) *n is at most 5 times the number of digits in b.*

Proof. TODO

Proposition 1.5.1. *The number of digits of a positive integer b is*

$$\lfloor \log_{10} b + 1 \rfloor$$

Proof. TODO

On analyzing the proof, the above is in fact true for any base > 1. In other words the number of base–B symbols to represent b is

$$\lfloor \log_B b + 1 \rfloor$$

where $B > 1$. For instance the number of bits needed to represent b is

$$\lfloor \log_2 b + 1 \rfloor$$

For instance, $b = 9_{10} = 1001_2$ which has 4 bits and

$$\lfloor \log_2 9 + 1 \rfloor = \lfloor 3.1699... + 1 \rfloor = \lfloor 4.1699... \rfloor = 4$$

Proposition 1.5.2. *Let positive integer b be written in base B where $B > 1$ is an integer. Then the number of base-B symbols used to represent b is*

$$\lfloor \log_B b + 1 \rfloor$$

Exercise 1.5.3. Leetcode 650.
https://leetcode.com/problems/2-keys-keyboard/
There is only one character `'A'` on the screen of a notepad. You can perform one of two operations on this notepad for each step:

- Copy All: You can copy all the characters present on the screen (a partial copy is not allowed).
- Paste: You can paste the characters which are copied last time.

Given an integer `n`, return the minimum number of operations to get the character `'A'` exactly `n` times on the screen.

Exercise 1.5.4. Leetcode 2447.
https://leetcode.com/problems/number-of-subarrays-with-gcd-equal-to-k
Given an integer array `nums` and an integer `k`, return the number of subarrays of `nums` where the greatest common divisor of the subarray's elements is `k`. A subarray is a contiguous non-empty sequence of elements within an array. The greatest common divisor of an array is the largest integer that evenly divides all the array elements.

Exercise 1.5.5. Leetcode 1998
https://leetcode.com/problems/gcd-sort-of-an-array/
You are given an integer array `nums`, and you can perform the following operation any number of times on `nums`:

- Swap the positions of two elements `nums[i]` and `nums[j]` if `gcd(nums[i], nums[j]) > 1`

where `gcd(nums[i], nums[j])` is the greatest common divisor of `nums[i]` and `nums[j]`.

Return true if it is possible to sort `nums` in non-decreasing order using the above swap method, or false otherwise.

1.6 Extended Euclidean algorithm: GCD as linear combination

Here's another super important fact:

Extended Euclidean Algorithm

Theorem 1.6.1. (**Extended Euclidean Algorithm**) *If a and b be integers which are not both zero, then there are integers x, y such that*

$$\gcd(a, b) = ax + by$$

Bézout's coefficients

The x, y in the above theorem are called **Bézout's coefficients** of a, b. They are not unique.

Exercise 1.6.1. Prove that $a \neq 0$, then there are many possible x, y such that $ax + by = \gcd(a, b)$. □

First let me prove that there are x, y such that

$$ax + by = \gcd(a, b)$$

The theorem does not give you the algorithm. Then I'll do a computational example that compute the gcd of a, b as a linear combination of a and b. The example actually contains the idea behind the algorithm to compute the Bézout's coefficients. The algorithm is called the Extended Euclidean Algorithm.

Proof. For convenience, let me write (a, b) as the set of linear combinations of a and b, i.e.,

$$(a, b) = \{ax + by \mid x, y \in \mathbb{Z}\}$$

We will also write (g) for the linear combination of g, i.e.,

$$(g) = \{gx \mid x \in \mathbb{Z}\}$$

(Such linear combinations are called ideals. They are extremely important in of themselves. Historically, they were created to study Fermat's last theorem. Since then they are crucial in the the study of ring theory.)

The proof proceeds in two steps:

1. Given a, b not both zero, there is some g such that $(a, b) = (g)$. If a, b are not both zero, g can be chosen to be > 0.
2. The g in the above is in fact $\gcd(a, b)$.

Now let's prove step 1, i.e., given a, b, there is some g such that

$$(a, b) = (g)$$

First of all if $b = 0$, then by definition

$$(a, 0) = (a)$$

and we're done. Next, we now assume $b \neq 0$. Then $|b| > 0$. The set

$$X = \{ax + by \mid x, y \in \mathbb{Z} \text{ and } ax + by > 0\} \subseteq \mathbb{N}$$

is nonempty since it contains $0 \cdot a + 1 \cdot |b|$. By the well-ordering principle of $\mathbb{N}$, X has a minimum element, say g. We will now show that $(a, b) = (g)$.

Since g is a minimum element of X, g is in X. Therefore $g = ax + by$. Hence $gz = a(xz) + b(yz) \in (a, b)$ for all $z \in \mathbb{Z}$. This implies that $(g) \subseteq (a, b)$.

Now we will prove that $(a, b) \subseteq (g)$. Let $c \in (a, b)$, i.e., $c = ax + by$ for some $x, y \in \mathbb{Z}$. Therefore by the Euclidean property of $\mathbb{Z}$, there exists $q, r \in \mathbb{Z}$ such that

$$c = gq + r, \;\; 0 \leq r < g$$

(Look at the definition of X again. X is a subset of $\mathbb{N}$ so that $g \geq 1$). If $r \neq 0$, then

$$r = c - gq$$

Note that $c = ax + by$ by our assumption. We have already shown that $(g) \subseteq (a, b)$, i.e., $g = ax' + by'$. Therefore, altogether we have

$$r = c - gq = ax + by - (ax' + b'y)q = a(x - x'q) + b(y - y'q)$$

Hence $r \in X$. But $0 \leq r < d$ implies that

$$r = a(x - x'q) + b(y - b'q)$$

is an element of X which is less than g which contradicts the minimality of g. Hence $r = 0$ and we have

$$c = gq + r = gq \in (g)$$

We have shown that $(a, b) \subseteq (g)$.

Altogether, we have shown $(a, b) = (g)$. Step 1 is now completed.

For step 2, we will show that g is the gcd of a and b. Since $(a, b) = (g)$, we have

$$a \in (a, b) = (g)$$

i.e., $a = xg$ which means g divides a. Likewise g divides b. Hence g is a common divisor of a and b. Since $(g) \subseteq (a, b)$, $g = ax_0 + by_0$. Suppose d is any divisor of a and b. Then $d \mid ax_0 + by_0$ by the linearity of divisibility. Hence $d \mid g$. Therefore g is the largest common divisor of a and b, i.e., $g = \gcd(a, b)$. □

The above does not give you an algorithm to compute the x and y. First let me do an example to show you that it's possible to compute $\gcd(a, b)$ as a linear combination of a and b. Then I'll give you the algorithm.

Recall that we have computed $\gcd(514, 24) = 2$. Extended Euclidean Algorithm says that it's possible to find x and y such that

$$2 = \gcd(514, 24) = 514x + 24y$$

How do we compute the x and y? Just like the gcd computation (the Euclidean Algorithm), the x, y are computed using the Euclidean property. First we have

$$514 = 21 \cdot 24 + 10$$

This implies that

$$514 \cdot 1 + 24 \cdot (-21) = 10$$

Now it would be nice if the pesky 10 goes away and is replaced by 2. How would we do that? Well look at 24 and 10 now. We have

$$24 = 2 \cdot 10 + 4$$

again by Euclidean algorithm. Multiplying the equation

$$514 \cdot 1 + 24 \cdot (-21) = 10$$

throughout by 2 gives us

$$514 \cdot 2 + 24 \cdot (-42) = 2 \cdot 10$$

The previous equation

$$24 = 2 \cdot 10 + 4$$

say that $2 \cdot 10$ can be replaced by $24 - 4$. This means that

$$514 \cdot 2 + 24 \cdot (-42) = 24 - 4$$

Hmmm ... this says that we have now

$$514 \cdot 2 + 24 \cdot (-43) = -4$$

or

$$514 \cdot (-2) + 24 \cdot 43 = 4$$

What about 4? Well, if we look at 10 and 4 just like what we did with 24 and 10 we would get

$$10 = 2 \cdot 4 + 2$$

and the remainder 2 gives us the GCD!!! Rearranging it a bit we have

$$1 \cdot 10 + (-2) \cdot 4 = 2$$

i.e. 2 is a linear combination of 10 and 4. But earlier we say that 4 is a linear combination of 514 and 24 ...

$$514 \cdot (-2) + 24 \cdot 43 = 4$$

and even earlier we saw that 10 is also a linear combination of 514 and 24 ...

$$514 \cdot 1 + 24 \cdot (-21) = 10$$

Surely if we substitute all these values into the equation

$$1 \cdot 10 + (-2) \cdot 4 = 2$$

we would get 2 as a linear combination of 514 and 24. Let's do it ...

$$\begin{aligned} 2 &= 1 \cdot 10 + (-2) \cdot 4 \\ &= 1 \cdot (514 \cdot 1 + 24 \cdot (-21)) + (-2)(514 \cdot (-2) + 24 \cdot 43) \\ &= 514 \cdot 1 + 24 \cdot (-21) + 514 \cdot 4 + 24 \cdot (-86) \\ &= 514 \cdot 5 + 24 \cdot (-107) \end{aligned}$$

Vóila!

Exercise 1.6.2. Using the above idea, compute the gcd and Bézout's coefficients of 210 and 78, i.e., compute x and y such that $210x+78y = \gcd(210, 78)$.

Exercise 1.6.3. Analyze the above and design an algorithm so that when given a and b, the algorithm computes x and y such that $ax + by = \gcd(a, b)$.

To help you analyze the above computation, let me organize our computations a little. If we can make the process systematic, then there is hope that we can make the idea work for all a and b, i.e., then we would have an algorithm and hence can program it and compute it's runtime performance.

We know for sure that we have to continually use Euclidean property on pairs of numbers. So here we go:

$$\begin{aligned} 514 &= 21 \cdot 24 + 10 \\ 24 &= 2 \cdot 10 + 4 \\ 10 &= 2 \cdot 4 + 2 \\ 4 &= 2 \cdot 2 + 0 \end{aligned}$$

Note that this corresponds to the gcd computation

$$\begin{aligned} \gcd(514, 24) &= \gcd(24, 514 - 21 \cdot 24) = \gcd(24, 10) \\ &= \gcd(10, 24 - 2 \cdot 10) = \gcd(10, 4) \\ &= \gcd(4, 10 - 2 \cdot 4) = \gcd(4, 2) \\ &= \gcd(2, 4 - 2 \cdot 2) = \gcd(2, 0) \\ &= 2 \end{aligned}$$

So in the computation

$$\begin{aligned} 514 &= 21 \cdot 24 + 10 \\ 24 &= 2 \cdot 10 + 4 \\ 10 &= 2 \cdot 4 + 2 \\ 4 &= 2 \cdot 2 + 0 \end{aligned}$$

if the remainder is 0 (see the last line), then the previous line's remainder must be the gcd.

Let's look at our computation of the gcd of 514 and 24:

$$\begin{aligned} 514 &= 21 \cdot 24 + 10 \\ 24 &= 2 \cdot 10 + 4 \\ 10 &= 2 \cdot 4 + 2 \\ 4 &= 2 \cdot 2 + 0 \end{aligned}$$

Recall that the above computation means that the gcd is 2. Note only that through backward substitution, we can rewrite 2 as a linear combination of 514 and 24.

Let's try to do this in a more organized way. So here's our facts again:

$$\begin{aligned} 514 &= 21 \cdot 24 + 10 \\ 24 &= 2 \cdot 10 + 4 \\ 10 &= 2 \cdot 4 + 2 \end{aligned}$$

Let me put the remainders on one side:

$$\begin{aligned} 10 &= 514 - 21 \cdot 24 && (1) \\ 4 &= 24 - 2 \cdot 10 && (2) \\ 2 &= 10 - 2 \cdot 4 && (3) \end{aligned}$$

Note that (1) tells you that 10 is a linear combination of $514, 24$. (2) tells you that 4 is a linear combination of $24, 10$. If we substitute (1) into (2), 4 will become a linear combination of $514, 24$. (3) says that 2 is a linear combination of $10, 4$. But 10 is a linear combination of $514, 24$ and 4 is a linear combination of $514, 24$. Hence 2 is also a linear combination of $514, 24$. See it?

OK. Let's do it. From

$$\begin{aligned} 10 &= 514 - 21 \cdot 24 && (1) \\ 4 &= 24 - 2 \cdot 10 && (2) \\ 2 &= 10 - 2 \cdot 4 && (3) \end{aligned}$$

if we substitute (1) into (2) and (3) (i.e., rewrite 10 as a linear combination of $514, 24$):

$$\begin{aligned} 10 &= 514 - 21 \cdot 24 && (1) \\ 4 &= 24 - 2 \cdot (514 - 21 \cdot 24) && (2) \\ 2 &= (514 - 21 \cdot 24) - 2 \cdot 4 && (3) \end{aligned}$$

Collecting the multiples of 514 and 24:

$$\begin{aligned} 10 &= 514 - 21 \cdot 24 && (1) \\ 4 &= (-2) \cdot 514 + (1 + (-2)(-21)) \cdot 24 && (2') \\ 2 &= (1) \cdot 514 + (-21) \cdot 24 - 2 \cdot 4 && (3') \end{aligned}$$

and simplifying:

$$\begin{aligned} 10 &= 514 - 21 \cdot 24 && (1)\\ 4 &= (-2) \cdot 514 + (43) \cdot 24 && (2')\\ 2 &= (1) \cdot 514 + (-21) \cdot 24 - 2 \cdot 4 && (3') \end{aligned}$$

Substituting $(2')$ into $(3')$:

$$\begin{aligned} 10 &= 514 - 21 \cdot 24 && (1)\\ 4 &= (-2) \cdot 514 + (43) \cdot 24 && (2')\\ 2 &= (1) \cdot 514 + (-21) \cdot 24 - 2 \cdot ((-2) \cdot 514 + (43) \cdot 24) && (3') \end{aligned}$$

Tidying up:

$$\begin{aligned} 10 &= 514 - 21 \cdot 24 && (1)\\ 4 &= (-2) \cdot 514 + (43) \cdot 24 && (2')\\ 2 &= (1 - 2(-2)) \cdot 514 + (-21 - 2(43)) \cdot 24 && (3'') \end{aligned}$$

Simplifying:

$$\begin{aligned} 10 &= 514 - 21 \cdot 24 && (1)\\ 4 &= (-2) \cdot 514 + (43) \cdot 24 && (2')\\ 2 &= (5) \cdot 514 + (-107) \cdot 24 && (3'') \end{aligned}$$

(It's a good idea to check after each substitution that the equalities still hold. We all make mistakes, right?)

OK. That's great. It looks more organized now. So much so that you can now easily write a program to compute the above.

Now let's look at the general case. Suppose instead of 514 and 24, we write a and b. The computation will look like this:

$$\begin{aligned} a &= q_1 \cdot b + r_1\\ b &= q_2 \cdot r_1 + r_2\\ r_1 &= q_3 \cdot r_2 + r_3\\ r_2 &= q_4 \cdot r_3 + 0 \end{aligned}$$

To make things even more regular and uniform, let me rewrite it this way:

$$\begin{aligned} r_0 &= q_1 \cdot r_1 + r_2 \\ r_1 &= q_2 \cdot r_2 + r_3 \\ r_2 &= q_3 \cdot r_3 + r_4 \\ r_3 &= q_4 \cdot r_4 + 0 \end{aligned}$$

A lot nicer, right? Let me write it this way with the remainder term on the lefts:

$$\begin{aligned} r_2 &= (1) \cdot r_0 + (-q_1) \cdot r_1 \\ r_3 &= (1) \cdot r_1 + (-q_2) \cdot r_2 \\ r_4 &= (1) \cdot r_2 + (-q_3) \cdot r_3 \end{aligned}$$

(Remember that r_4 is the gcd ... $r_0 = 514, r_1 = 24$ ... right?) Organized this way, we have the gcd on one side of the equation. Now if we substitute the first equation into the second we get

$$\begin{aligned} r_2 &= (1) \cdot r_0 + (-q_1) \cdot r_1 \text{ ... USED} \\ r_3 &= (1) \cdot r_1 + (-q_2) \cdot ((1) \cdot r_0 + (-q_1) \cdot r_1) \\ r_4 &= (1) \cdot r_2 + (-q_3) \cdot r_3 \end{aligned}$$

i.e.,

$$\begin{aligned} r_2 &= (1) \cdot r_0 + (-q_1) \cdot r_1 \text{ ... USED} \\ r_3 &= (-q_2) \cdot r_0 + (1 + q_1 q_2) \cdot r_1 \\ r_4 &= (1) \cdot r_2 + (-q_3) \cdot r_3 \end{aligned}$$

Note that we cannot throw away the first equation yet. We need to keep r_2 around since it appears in the third equation! So when can we throw the first equation away? Look at the general case. Suppose we have

$$\begin{aligned} r_2 &= (1) \cdot r_0 + (-q_1) \cdot r_1 \\ r_3 &= (1) \cdot r_1 + (-q_2) \cdot r_2 \\ r_4 &= (1) \cdot r_2 + (-q_3) \cdot r_3 \\ r_5 &= (1) \cdot r_3 + (-q_4) \cdot r_4 \\ r_6 &= (1) \cdot r_4 + (-q_5) \cdot r_5 \\ &\ldots \end{aligned}$$

Aha! r_2 is used only in the next *two* equations.

Suppose we are at equation 3:

$$r_3 = c_1 \cdot r_0 + d_1 \cdot r_1$$
$$r_4 = c_2 \cdot r_0 + d_2 \cdot r_1$$

We have to compute the next equation: This requires r_3, r_4. Then we have

$$r_5 = (1) \cdot r_3 + (-q_4) \cdot r_4$$

where

$$q_4 = \lfloor r_3/r_4 \rfloor, \quad r_5 = r_3 - q_4 r_4$$

Altogether we have

$$r_3 = c_1 \cdot r_0 + d_1 \cdot r_1$$
$$r_4 = c_2 \cdot r_0 + d_2 \cdot r_1$$
$$r_5 = (1) \cdot r_3 + (-q_4) \cdot r_4$$

The last equation becomes

$$r_5 = c_1 \cdot r_0 + d_1 \cdot r_1 + (-q_4) \cdot (c_2 \cdot r_0 + d_2 \cdot r_1)$$

i.e.

$$r_5 = (c_1 - q_4 c_2) \cdot r_0 + (d_1 - q_4 d_2) \cdot r_1$$

Let me repeat that in a slightly more general context. If we have

$$r_3 = c_1 \cdot r_0 + d_1 \cdot r_1$$
$$r_4 = c_2 \cdot r_0 + d_2 \cdot r_1$$

then we get (throwing away the first equation):

$$r_4 = c_2 \cdot r_0 + d_2 \cdot r_1$$
$$r_5 = (c_1 - q_4 c_2) \cdot r_0 + (d_1 - q_4 d_2) \cdot r_1$$

To put it in terms of numbers instead of equations this is what we get: If we have

$$c_1, d_1, c_2, d_2, r_3, r_4$$

then we get

$$c_2, d_2, c_1 - \lfloor r_3/r_4 \rfloor c_2, d_1 - \lfloor r_3/r_4 \rfloor d_2, r_4, r_3 - \lfloor r_3/r_4 \rfloor r_4$$

In general, if we have

$$c, d, c', d', r, r'$$

then we get

$$c', d', c - \lfloor r/r' \rfloor\, c', d - \lfloor r/r' \rfloor\, d', r', r - \lfloor r/r' \rfloor\, r'$$

Of course since we start off with r_0, r_1 (i.e. what we call a and b above), we have

$$\begin{aligned} r_0 &= 1 \cdot r_0 + 0 \cdot r_1 \\ r_1 &= 0 \cdot r_0 + 1 \cdot r_1 \end{aligned}$$

i.e., you would start off with

$$c = 1, d = 0, c' = 0, d' = 1, r = r_0, r' = r_1$$

Let's check this algorithm with our $r_0 = 514, r_1 = 24$.

STEP 1: The initial numbers are

$$c = 1, d = 0, c' = 0, d' = 1, r = 514, r' = 24$$

Again this corresponds to

$$\begin{aligned} r_3 &= 1 \cdot 514 + 0 \cdot 24 \\ r_4 &= 0 \cdot 514 + 1 \cdot 24 \end{aligned}$$

STEP 2: The new numbers (6 of them) are

$$\begin{aligned} c' &= 0 \\ d' &= 1 \\ c - \lfloor r/r' \rfloor\, c' &= 1 - \lfloor 514/24 \rfloor\, 0 = 1 \\ d - \lfloor r/r' \rfloor\, d' &= 0 - \lfloor 514/24 \rfloor\, 1 = 0 - 21 = -21 \\ r' &= 24 \\ r - \lfloor r/r' \rfloor\, r' &= 514 - \lfloor 514/24 \rfloor\, 24 = 514 - 504 = 10 \end{aligned}$$

So the new numbers (we reset the variables in the algorithm):

$$c = 0, d = 1, c' = 1, d' = -21, r = 24, r' = 10$$

These corresponds to the data on the second and third line of the following:

$$\begin{aligned} 514 &= 1 \cdot 514 + 0 \cdot 24 \\ 24 &= 0 \cdot 514 + 1 \cdot 24 \\ 10 &= 1 \cdot 514 + (-21) \cdot 24 \end{aligned}$$

STEP 3: From the 6 numbers from STEP 2 we get

$$\begin{aligned} c' &= 1 \\ d' &= -21 \\ c - \lfloor r/r' \rfloor \, c' &= 0 - \lfloor 24/10 \rfloor \, 1 = -2 \\ d - \lfloor r/r' \rfloor \, d' &= 1 - \lfloor 24/10 \rfloor \, (-21) = 1 + 42 = 43 \\ r' &= 10 \\ r - \lfloor r/r' \rfloor \, r' &= 24 - \lfloor 24/10 \rfloor \, 10 = 24 - 20 = 4 \end{aligned}$$

So the new numbers (we reset the variables in the algorithm):

$$c = 1, d = -21, c' = -2, d' = 43, r = 10, r' = 4$$

These corresponds to the data on the third and fourth line of the following:

$$\begin{aligned} 514 &= 1 \cdot 514 + 0 \cdot 24 \\ 24 &= 0 \cdot 514 + 1 \cdot 24 \\ 10 &= 1 \cdot 514 + (-21) \cdot 24 \\ 4 &= (-2) \cdot 514 + 43 \cdot 24 \end{aligned}$$

STEP 4: From the 6 numbers from STEP 3 we get

$$\begin{aligned} c' &= -2 \\ d' &= 43 \\ c - \lfloor r/r' \rfloor \, c' &= 1 - \lfloor 10/4 \rfloor \, (-2) = 1 + 4 = 5 \\ d - \lfloor r/r' \rfloor \, d' &= -21 - \lfloor 10/4 \rfloor \, (43) = -21 - 86 = -107 \\ r' &= 4 \\ r - \lfloor r/r' \rfloor \, r' &= 10 - \lfloor 10/4 \rfloor \, 4 = 10 - 8 = 2 \end{aligned}$$

So the new numbers (we reset the variables in the algorithm):

$$c = -2, d = 43, c' = 5, d' = -107, r = 4, r' = 2$$

These corresponds to the data on the fourth and fifth line of the following:

$$\begin{aligned} 514 &= 1 \cdot 514 + 0 \cdot 24 \\ 24 &= 0 \cdot 514 + 1 \cdot 24 \\ 10 &= 1 \cdot 514 + (-21) \cdot 24 \\ 4 &= (-2) \cdot 514 + 43 \cdot 24 \\ 2 &= 5 \cdot 514 + (-107) \cdot 24 \end{aligned}$$

STEP 5: From the 6 numbers from STEP 4 we get

$$\begin{aligned} c' &= 5 \\ d' &= -107 \\ c - \lfloor r/r' \rfloor\, c' &= -2 - \lfloor 4/2 \rfloor\, 5 = -2 - 10 = -12 \\ d - \lfloor r/r' \rfloor\, d' &= 43 - \lfloor 4/2 \rfloor\, (-107) = 43 + 214 = 257 \\ r' &= 2 \\ r - \lfloor r/r' \rfloor\, r' &= 4 - \lfloor 4/2 \rfloor\, 2 = 4 - 4 = 0 \end{aligned}$$

So the new numbers (we reset the variables in the algorithm):

$$c = 5, d = -107, c' = -12, d' = 257, r = 2, r' = 0$$

These corresponds to the data on the fifth and sixth line of the following:

$$\begin{aligned} 514 &= 1 \cdot 514 + 0 \cdot 24 \\ 24 &= 0 \cdot 514 + 1 \cdot 24 \\ 10 &= 1 \cdot 514 + (-21) \cdot 24 \\ 4 &= (-2) \cdot 514 + 43 \cdot 24 \\ 2 &= 5 \cdot 514 + (-107) \cdot 24 \\ 0 &= (-12) \cdot 514 + 257 \cdot 24 \end{aligned}$$

Of course (as before) at this point, you see that the $r' = 0$. Therefore

$$\gcd(514, 24) = 2$$

and furthermore from $c = 5, d = -107$, we get

$$5 \cdot 514 + (-107) \cdot 24 = \gcd(514, 24)$$

Here's a Python implementation with some test code:

```
ALGORITHM: EEA
INPUTS: a, b
OUTPUTS: r, c, d where r = gcd(a, b) = c*a + d*b

a0, b0 = a, b
d0, d = 0, 1
c0, c = 1, 0
q = a0 // b0
r = a0 - q * b0

while r > 0:
    d, d0 = d0 - q * d, d
    c, c0 = c0 - q * c, c

    a0, b0 = b0, r
    q = a0 // b0
    r = a0 - q * b0

r = b0
return r, c, d
```

You can pound real hard at the Extended Euclidean Algorithm with this:

By the way, this is somewhat similar to what we call *tail recursion* (CISS445) an extremely important technique in functional programming. All LISP hackers and people working in high performance computing and compilers swear by it. You don't see recursion in the above code, but you can replace the while-loop with recursion and if you have a compiler/interpreter that can perform true tail recursion, then it would run exactly like the above algorithm.

Exercise 1.6.4. Leetcode 365
https://leetcode.com/problems/water-and-jug-problem/description/
(Also, Die Hard 3 problem https://www.math.tamu.edu/~dallen/hollywood/diehard/diehard.htm.) You are given two jugs with capacities `jug1Capacity` and `jug2Capacity` liters. There is an infinite amount of water supply available. Determine whether it is possible to measure exactly `targetCapacity`

liter using these two jugs.

If `targetCapacity` liters of water are measurable, you must have `targetCapacity` liters of water contained within one or both buckets by the end.

Operations allowed:

1. Fill any of the jugs with water.
2. Empty any of the jugs.
3. Pour water from one jug into another till the other jug is completely full, or the first jug itself is empty.

You'll see that there are times when you're only interested in the value of x and not y (or y and not x – the above is symmetric about x and y). Do you notice x comes from c? If you analyze the above algorithm, you see immediately that c is computed from c' and c' is computed from c, c', q, q is computed from r, r', r is computed from r', and finally (phew!) r' is computed from r, q, r'. Therefore if you're interested in c, you don't need to compute d or d'. So you can change the EEA to this:

```
ALGORTHM: EEA2 (sort of EEA ... without the d, d0)
INPUTS: a, b
OUTPUTS: r, c where r = gcd(a, b) = c*a + d*b for some d

a0, b0 = a, b
c0, c = 1, 0
q = a0 // b0
r = a0 - q * b0

while r > 0:
    c, c0 = c0 - q * c, c

    a0, b0 = b0, r
    q = a0 // b0
    r = a0 - q * b0

r = b0
return r, c
```

Later you'll see why we compute only c. It's not that we have something against d.

Exercise 1.6.5. Compute the following gcd and the Bézout's coefficients of the given numbers by following the Extended Euclidean Algorithm.

1. $\gcd(0, 10)$
2. $\gcd(10, 0)$
3. $\gcd(10, 1)$
4. $\gcd(10, 10)$
5. $\gcd(107, 5)$
6. $\gcd(107, 26)$
7. $\gcd(84, 333)$
8. $\gcd(F_{10}, F_{11})$ where F_n is the n–th Fibonacci number. (Recall: $F_0 = 0, F_1 = 1, F_{n+2} = F_{n+1} + F_n$.)
9. $\gcd(ab, b)$
10. $\gcd(a, a + 1)$
11. $\gcd(ab + a, b)$ where $0 < a < b$. Go as far as you can.
12. $\gcd(a(a + 1) + a, (a + 1))$ where $0 < a < b$. Go as far as you can.

Exercise 1.6.6. Prove that if $a \mid c$, $b \mid c$, and $\gcd(a, b) = 1$, then $ab \mid c$.

Exercise 1.6.7. Prove that if $a \mid c$, $b \mid c$, then

$$\frac{ab}{\gcd(a, b)} \mid c$$

Exercise 1.6.8. Leetcode 920.
https://leetcode.com/problems/number-of-music-playlists
Your music player contains n different songs. You want to listen to goal songs (not necessarily different) during your trip. To avoid boredom, you will create a playlist so that:

- Every song is played at least once.
- A song can only be played again only if k other songs have been played.

Given `n`, `goal`, and `k`, return the number of possible playlists that you can create. Since the answer can be very large, return it modulo $109 + 7$.

1.7 Primes

Definition 1.7.1. A prime p is a positive integer greater than 1 that is divisible by only 1 and itself. In other words $p \in \mathbb{N}$ is a **prime** if $p > 1$ and if $d \mid p$, then $d = 1$ or $d = p$. prime

Examples of primes are $2, 3, 5, 7, 11, 13, 17, 19, ...$

Integers least zero can be divided into the following types:

- 0 – the zero element
- 1 – the unit element (i.e. the only invertible element ≥ 0)
- primes – $2, 3, 5, 7, 11, ...$
- **composites** – integers > 1 which are not primes composites

(Instead of primes of $\mathbb{N} \cup \{0\}$, it's also possible to define primes of $\mathbb{Z}$. A prime of $\mathbb{Z}$ is an integer not $-1, 0, 1$ such that if $d \mid p$, then $d = \pm 1$ or $d = \pm p$. In that case primes of $\mathbb{Z}$ are $\pm 2, \pm 3, \pm 5, \pm 7, \pm 11, ...$)

Proposition 1.7.1. (Euclid) *There are infinitely many primes.*

Proof. TODO

Proposition 1.7.2. (Euclid) *There are arbitrarily long consecutives integers of composites.*

Proof. $n! + 2$, $n! + 3$, $n! + 4$, ..., $n! + n$ are all composities for $n \geq 2$. This list is therefore a list of consecutive composites of length $n - 1$. □

The follow lemma is extremely important and is used for instance in the fundamental theorem of arithmetic to prove the uniqueness of prime factorization in $\mathbb{N}$ (or $\mathbb{Z}$). To break tradition, we will call this a theorem (instead of a lemma):

Theorem 1.7.1. (**Euclid's lemma**) *If p is a prime and $p \mid ab$, then either $p \mid a$ or $p \mid b$.* Euclid's lemma

Proof. TODO

The above generalizes easily (by induction) to the following:

Corollary 1.7.1. *If p is a prime and $p \mid a_1 a_2 \cdots a_n$, then p divides at least one of the $a_1, \ldots, a_n$.*

Proof. We will prove this by strong induction on the number of terms in $a_1, ..., a_n$. Hence let $P(n)$ be the above statement, i.e., $P(n)$ is the statement that if a prime divides the product of n terms, then p divides at least one of the terms.

The case of $n = 2$ is Euclid's lemma. Hence $P(2)$ holds. This is the base case of our induction.

Now assume $P(k)$ is true for $2 \leq k \leq n$. Let p be a prime divide $a_1 a_2 \cdots a_n a_{n+1}$. Therefore p divides $a_1 a_2 \cdots a_{n-1} b$ (there are $n-1$ terms) where $b = a_n a_{n+1}$. Since $P(n)$ is true, p divides at least one of $a_1, ..., a_{n-1}, b$. Since $P(2)$ is true, p divides $b = a_n a_{n+1}$ if p divides a_n or a_{n+1}. Hence p divides at least one of $a_1, ..., a_{n+1}$. Therefore $P(n+1)$ holds. □

Fundamental Theorem of Arithmetic

Theorem 1.7.2. (Euclid's **Fundamental Theorem of Arithmetic**) *Every positive integer > 1 can be written as a unique product of primes up to permutation of the prime factors. This means*

(a) If $n > 1$ is an integer, then n can be written as a product of primes.
(b) If n is written as two products of primes:

$$n = p_1 p_2 \cdots p_k = q_1 q_2 \cdots q_\ell$$

where p_i and q_j are primes arranged in ascending order, i.e.,

$$p_1 \leq p_2 \leq \cdots \leq p_k$$
$$q_1 \leq q_2 \leq \cdots \leq q_\ell$$

then $k = \ell$ and

$$p_1 = q_1, \quad p_2 = q_2, \quad \cdots, \quad p_k = q_k,$$

Proof. TODO

The statement of the Fundamental Theorem of Arithmetic can include the case of $n = 1$ if we accept that the product of an empty set of integers is 1:

$$\prod_{p \in \{\}} p = 1$$

Proposition 1.7.3. *Let* $a = \prod_{p \in P} p^{a_p}$, $b = \prod_{p \in P} p^{b_p}$ *and* $c = \prod_{p \in P} p^{c_p}$ *where* P *is a finite set of primes. Then*

(a) $c = ab \implies c_p = a_p + b_p$.
(b) $a \mid b \implies a_p \leq b_p$ *for all* $p \in P$.
(c) $c = \gcd(a, b) \implies c_p = \min(a_p, b_p)$.

The above assumes the easily proven facts that

$$\prod_{p \in P} p^{a_p} \prod_{p \in P} p^{b_p} = \prod_{p \in P} p^{a_p + b_p}$$

(by commutativity of $\cdot$) and

$$\prod_{p \in P} p^{a_p} = \prod_{p \in P} p^{b_p} \implies a_p = b_p \text{ for all } p \in P$$

by uniqueness of prime factorization from the Fundamental Theorem of Arithmetic. Here, P is a set of distinct primes.

Proposition 1.7.4. *If* $n > 1$ *is not a prime, then there is a prime factor* p *such that* $p \leq \sqrt{n}$.

Proof. TODO □

Therefore a very simple primality test algorithm for n is the following:

```
ALGORITHM: BRUTE-FORCE-PRIMALITY-TEST
INPUT: n
OUTPUT: true if n is prime. If n < 2, false is returned.

if n < 2: return false

d = 2
while d <= sqrt(n):
    if n % d == 0:
        return false
    d = d + 1
return true
```

In terms of n, the runtime is $O(\sqrt{n})$. However in terms of the bits of n, by Proposition 1.5.2, the number of n is

$$b = \lfloor \log_2(n+1) \rfloor = \log_2(n+1) - \alpha$$

where $0 \leq \alpha < 1$. Hence

$$n = 2^b 2^\alpha - 1$$

Hence in terms of the number of bits of n, the runtime is

$$O(\sqrt{n}) = O(\sqrt{2^b 2^\alpha - 1}) = O(2^{b/2})$$

It is common to denote the number of bits of the input by n. Hence the runtime in number of bits n is $O(2^{n/2})$, i.e., it has **exponential runtime with linear exponent**.

exponential runtime with linear exponent

$O(\sqrt{n})$ is said to be the **pseudo-polynomial runtime** of the algorithm to indicate that the n is the numeric input and not a correct measure of the complexity of the input, which should be in number of bits of the input.

pseudo-polynomial runtime

Exercise 1.7.1. Let $P = \{p_1, ..., p_n\}$ be a set of distint primes. Consider the expression $N(P) = \prod_{p \in P} p + 1$ in Euclid's proof of infinitude of primes. This is sometimes called Euclid's construction. How often is this a prime?

Exercise 1.7.2. Let $P = \{p_1, ..., p_n\}$ be a set of distint primes. Carry out the Euclid's construction on all possible subsets of P. If a Euclid construction is a prime, put that prime into P. If it does not, put the smallest prime factor into P. Repeat. For instance if you start with $P = \{\}$, you'll get 2 and the new P is $\{2\}$. Next you'll get $P = \{2, 3\}$. The Euclid constructions you get from P are $2, 3, 4, 7$. So the next P is $\{2, 3, 7\}$. At the next stage you get $2, 3, 4, 8, 7, 15, 22, 43$. This means that the next P is $\{2, 3, 7, 43\}$. Etc. Does your P always grow? Notice that your P's so far does not capture 5. When, if at all, will 5 appear?

Exercise 1.7.3. In the above, if an Euclid construction does not give you a prime, you take the smallest prime factor. What if you pick the largest prime factor?

The following are some DIY exercises for self-study on famous unsolved problems in number theory. You might want to write programs to check on the conjectures.

Exercise 1.7.4. Are there infinitely many primes p such that $p + 2$ is also a

prime? If p and $p+2$ are both primes, then they are called **twin primes**. The **twin prime conjecture** states that there are infinitely many twin primes. (See https://en.wikipedia.org/wiki/Twin_prime.) Write a program that prints $p, p+2$ if both are primes. Print the time elapsed between the discovery of pairs of twin primes.

twin primes

twin prime conjecture

Exercise 1.7.5. Can even positive integer be written as the sum of two primes? When the two primes are > 2, then the sum of these two primes is even. The **Goldbach conjecture** states that every positive integer can be written as the sum of two primes. (See https://en.wikipedia.org/wiki/Goldbach%27s_conjecture.) Write a program that prints $n \geq 2$ as a sum of two primes as n iterates from 2 to a huge positive integer N entered by the user.

Goldbach conjecture

Exercise 1.7.6. A positive integer is a **perfect** number if it is the sum of its positive divisors strictly less than itself. For instance 6 is perfect since $6 = 1 + 2 + 3$. 28 is also perfect since $28 = 1 + 2 + 4 + 7 + 14$. Euclid knew that if p is prime and $2^p - 1$ is a prime, then $2^{p-1}(2^p - 1)$ is an even perfect number. If p is a prime and $2^p - 1$ is a also a prime, then $2^p - 1$ is called a **Mersenne prime**. Almost 2000 years after Euclid, Euler proved the converse, that every even perfect number must be of the form $2^{p-1}(2^p - 1)$ where $2^p - 1$ is a Mersenne prime. There are two famous unsolved problems in number theory on perfect numbers:

1. Are there odd perfect numbers?
2. Are there infinitely many perfect numbers?

(See https://en.wikipedia.org/wiki/Perfect_number.) There is an ongoing search for primes and Mersenne primes using computers. At this point (2023), the top 8 largest known primes are all Mersenne primes. (See https://en.wikipedia.org/wiki/Great_Internet_Mersenne_Prime_Search.) As of Feb 2023, the largest known prime is $2^{82,589,933} - 1$. (See https://en.wikipedia.org/wiki/Largest_known_prime_number.) Write a program that prints perfect numbers, printing the time between pairs of perfect numbers discovered.

Exercise 1.7.7. Leetcode 204
https://leetcode.com/problems/count-primes/
Given an integer n, return the number of prime numbers that are strictly less

than n.

Exercise 1.7.8. Leetcode 507
https://leetcode.com/problems/perfect-number/
A perfect number is a positive integer that is equal to the sum of its positive divisors, excluding the number itself. A divisor of an integer `x` is an integer that can divide `x` evenly. Given an integer `n`, return `true` if `n` is a perfect number, otherwise return `false`.

Exercise 1.7.9. Leetcode 866
https://leetcode.com/problems/prime-palindrome/
Given an integer n, return the smallest prime palindrome greater than or equal to n. An integer is prime if it has exactly two divisors: 1 and itself. Note that 1 is not a prime number. For example, 2, 3, 5, 7, 11, and 13 are all primes. An integer is a palindrome if it reads the same from left to right as it does from right to left. For example, 101 and 12321 are palindromes. The test cases are generated so that the answer always exists and is in the range $[2, 2 * 108]$.

Exercise 1.7.10. Leetcode 1175
https://leetcode.com/problems/prime-arrangements/ Return the number of permutations of 1 to n so that prime numbers are at prime indices (1-indexed.) (Recall that an integer is prime if and only if it is greater than 1, and cannot be written as a product of two positive integers both smaller than it.) Since the answer may be large, return the answer modulo $10^9 + 7$.

Exercise 1.7.11. Leetcode 1362
https://leetcode.com/problems/closest-divisors/
Given an integer `num`, find the closest two integers in absolute difference whose product equals `num + 1` or `num + 2`. Return the two integers in any order.

Exercise 1.7.12. Leetcode 1390
https://leetcode.com/problems/four-divisors/
Given an integer array `nums`, return the sum of divisors of the integers in that array that have exactly four divisors. If there is no such integer in the array, return `0`.

Exercise 1.7.13. Leetcode 2523
https://leetcode.com/problems/closest-prime-numbers-in-range/
Given two positive integers `left` and `right`, find the two integers `num1` and `num2` such that:

- `left <= nums1 < nums2 <= right`
- `nums1` and `nums2` are both prime numbers.
- `nums2 - nums1` is the minimum amongst all other pairs satisfying the above conditions.

Return the positive integer array `ans = [nums1, nums2]`. If there are multiple pairs satisfying these conditions, return the one with the minimum nums1 value or `[-1, -1]` if such numbers do not exist. A number greater than 1 is called prime if it is only divisible by 1 and itself.

Exercise 1.7.14. Leetcode 2521
https://leetcode.com/problems/distinct-prime-factors-of-product-of-array/
Given an array of positive integers `nums`, return the number of distinct prime factors in the product of the elements of `nums`.

Exercise 1.7.15. Leetcode 1071
https://leetcode.com/problems/greatest-common-divisor-of-strings/
For two strings s and t, we say "t divides s" if and only if $s = t + \cdots + t$ (i.e., t is concatenated with itself one or more times). Given two strings `str1` and `str2`, return the largest string `x` such that `x` divides both `str1` and `str2`.

1.8 Multiplicative inverse in $\mathbb{Z}/N$

Quick review: In algebra classes, lots of time is spent on solving equations. You usually start with something like: "Find the roots of $x + b$". This means finding a value for x such that

$$x + a = 0$$

Of course this is dead easy. As long as you have the concept of $-a$ (additive inverse) you just add $-a$ to both sides and voila:

$$\begin{aligned}(x + a) + (-a) &= 0 + (-a)\\ x + (a + (-a)) &= 0 + (-a)\\ x + 0 &= 0 + (-a)\\ x &= 0 + (-a)\\ x &= -a\end{aligned}$$

Next up, you usually see this: "Find the roots of $ax + b$" which is the same as finding a value for x such that

$$ax + b = 0$$

Assuming you are working in $\mathbb{R}$, you would do something like this:

$$\begin{aligned}(ax + b) + (-b) &= 0 + (-b)\\ ax + (b + (-b)) &= 0 + (-b)\\ ax + 0 &= 0 + (-b)\\ ax &= 0 + (-b)\\ ax &= -b\end{aligned}$$

and at this point you would do this:

$$\begin{aligned}ax &= -b\\ a^{-1}(ax) &= a^{-1}(-b)\\ (a^{-1}a)x &= a^{-1}(-b)\\ 1x &= a^{-1}(-b)\\ x &= a^{-1}(-b)\end{aligned}$$

and vóila (again) and you're done.

In the case of real numbers (or even just fractions), if you're given a number a

(which is not zero), you always have another number called a^{-1} (multiplicative inverse) such that

$$a \cdot a^{-1} = 1 = a^{-1} \cdot a$$

The reason why we like this is exactly because it allows us to solve the above equation.

Given a number a, the number $-a$ is called an **additive inverse** if it behaves like this: additive inverse

$$a + (-a) = 0 = (-a) + a$$

The number a^{-1} is called a **multiplicative inverse** of a if it does this: multiplicative inverse

$$a \cdot a^{-1} = 1 = a^{-1}a$$

If a has a multiplicative inverse, we also say that a is invertible.

Definition 1.8.1. We also say that a is a **unit** if a has a multiplicative inverse. unit

Definition 1.8.2. The set of multiplicatively invertible elements of $\mathbb{Z}/N$ is denoted by $(\mathbb{Z}/N)^\times$ or $U(\mathbb{Z}/N)$.

Almost all values in $\mathbb{R}$ are invertible; the only exception being 0. On the other hand, note that most numbers in $\mathbb{Z}$ do not have multiplicative inverses. In fact the only numbers in $\mathbb{Z}$ that have inverses are 1 and -1. So the only units in $\mathbb{Z}$ are $1, -1$.

Proposition 1.8.1. *The only units of $\mathbb{Z}$ are 1, -1. In particular, $1^{-1} = 1$ and $(-1)^{-1} = -1$.*

Proof. The fact $1 \cdot 1 = 1$ follows from the neutral axiom of $\cdot$. Hence $1^{-1} = 1$. $(-1)(-1) = 1$ is from Proposition 1.5.1(d). Hence $(-1)^{-1} = 1$.

Let $a \in \mathbb{Z}$ be invertible. Then $a \cdot a^{-1} = 1$. If $a = 0$, then $a \cdot a^{-1} = 1$ and $a \cdot a^{-1} = 0 \cdot a^{-1} = 0$. In other words $0 = 1$. But this contradicts the nontriviality axiom of $\mathbb{Z}$. Hence $a \neq 0$. Either $a > 0$ or $a < 0$.

First suppose $a > 0$. Note that $a^{-1} > 0$, otherwise $a \cdot a^{-1} < 0$ which contradicts $a \cdot a^{-1} = 1$. Since $a > 0$ and $a^{-1} > 0$, they have prime factorizations. Let p be a prime. Suppose the p–power appearing in the prime factorization of a is p^α

where $\alpha \geq 0$ and the p–power appearing in the prime factorization of a^{-1} is p^{β} where $\alpha \geq 0$. Then the p–power that appears in $a \cdot a^{-1}$ is $p^{\alpha+\beta}$. Note that $a \cdot a^{-1} = 1$ and the p–power that appears in the prime factorization of 1 is p^0. Therefore $p^{\alpha+\beta} = p^0$. Since $\alpha = 0 = \beta$. This implies that p does not appear in the prime factorization of a nor a^{-1}. Since this holds for all primes p, the prime factorization of a is $\prod_{p \in \emptyset} p$, i.e., $a = 1$.

Now suppose $a < 0$. Then $-a > 0$. Then the above case applied to $-a$ implies that $-a = 1$. Hence $-(-a) = -1$, i.e., $a = -1$ by Proposition 1.5.1(b). Hence $a = -1 = a^{-1}$.

Therefore the only units of $\mathbb{Z}$ are $1, -1$. □

Recall from the section on congruences, we have a relation $\equiv \pmod{N}$ that relates values of $\mathbb{Z}$. For instance when $N = 26$,

$$3 \equiv 29 \pmod{26}$$

and

$$3 \equiv -26 \pmod{26}$$

You can think of $\equiv \pmod{26}$ as a kind of equality on $\mathbb{Z}$ so that, in mod 26, 3 and 29 are the same and 3 and -26 are the same. In this context, write $\mathbb{Z}/N$ for the set

$$\mathbb{Z}/N = \{0, 1, 2, ..., N-1\}$$

We call the set $\mathbb{Z}/\mathbb{N}$ "$\mathbb{Z}$ mod N". Note that this set $\mathbb{Z}/N$ also contains N, but N is "the same as" 0, i.e., N is just another way to write 1 in the sense that

$$N \equiv 1 \pmod{N}$$

Note that $\mathbb{Z}/N$ also has $+$, $\cdot$ and has 0 and 1.

In fact $(\mathbb{Z}/N, +, \cdot, 0, 1)$ satisfies the algebraic properties of $+$, properties of $\cdot$, and distributivity. If $N > 1$, $(\mathbb{Z}/N, +, \cdot, 0, 1)$ also satisfies the nontriviality axiom, i.e., $0 \not\equiv 1 \pmod{N}$. However $(\mathbb{Z}/N, +, \cdot, 0, 1)$ does not satisfy the integrality axiom. Furthermore there's no concept of order $<$ on $\mathbb{Z}/N$ and hence there's no WOP or induction on $\mathbb{Z}/N$.

Later we will define the concept of commutative rings which are exactly sets, each with its own $+$ and $\cdot$ operations, satisfying the properties of $+$ and $\cdot$ and distributivity of $\mathbb{Z}$. $\mathbb{Z}$ and $\mathbb{Z}/N$ are both commutative rings.

The ability for a number in some commutative ring to have a multiplicative inverse (in that ring) is special.

On the other hand, $\mathbb{Z}/N$ ($N > 1$) might contain lots of units. For instance look at $\mathbb{Z}/10$. Is 3 invertible mod 10? In other words is there some x such that

$$3x \equiv 1 \pmod{10}$$

$x \equiv 7 \pmod{10}$ satisfies the above. In other words in mod 10, 7 is the multiplicative inverse of 3.

(You can talk about $\mathbb{Z}/N$ for $N = 1$, but it's not very interesting nor useful. $\mathbb{Z}/1$ has only one value, i.e., 0 which behaves like the additive identity element 0 and the multiplicative identity element 1.)

Here's the obvious brute algorithm to find the multiplicative inverse of an integer mod N:

```
ALGORITHM: brute-force-inverse
INPUT: a, N
OUTPUT: x such that a * x % N is 1

for x = 1, 2, 3, ..., N - 1:
    if a * x % N == 1:
        return x
return None
```

Note that in the above algorithm we need not test 0.

Exercise 1.8.1.

(a) Find all the units and their multiplicative inverses in $\mathbb{Z}/10$.
(b) Find all the units and their multiplicative inverses in $\mathbb{Z}/5$.
(c) Find all the units and their multiplicative inverses in $\mathbb{Z}/6$.

Do you see a pattern? □

It turns out that in $\mathbb{Z}/N$, you can easily find all the elements with multiplicative inverses: $a \in \mathbb{Z}/N$ has a multiplicative inverse if $\gcd(a, N) = 1$.

Proposition 1.8.2. *Let a, N be integers where $N > 0$. Then a is (multiplicatively) invertible in $\mathbb{Z}/N$ iff* $\gcd(a, N) = 1$.

If m, n are two integers such that $\gcd(m, n) = 1$, we say that m and n are **coprime**.

coprime

Proof. TODO □

The above immediately implies that

$$\begin{aligned}(\mathbb{Z}/N)^\times &= \{a \mid 0 \le a \le N-1,\ \ a \text{ is invertible mod } N\} \\ &= \{a \mid 0 \le a \le N-1,\ \ \gcd(a,N) = 1\}\end{aligned}$$

(Of course $\gcd(0, N) = N > 0$, so we can throw away 0 in the above if $N > 1$.) So the key is the computation of x, y such that

$$1 = \gcd(a, N) = ax + Ny$$

which is achieved by the Extended Euclidean Algorithm. But wait a minute ... I just need the x. So I'll just modify the Extended Euclidean Algorithm slightly.

Here's the (earlier) EEA algorithm, slightly modified, together with a function to compute the multiplicative inverse of $a \pmod N$:

```
ALGORITHM: EEA2 (sort of EEA ... without the d, d0)
INPUTS: a, b
OUTPUTS:  r, c where r = gcd(a, b) = c*a + d*b for some d

a0, b0 = a, b
c0, c = 1, 0
q = a0 // b0
r = a0 - q * b0

while r > 0:
    c, c0 = c0 - q * c, c

    a0, b0 = b0, r
    q = a0 // b0
    r = a0 - q * b0

r = b0
return r, c


ALGORITHM: mod-inverse
INPUTS: a, N
OUTPUT: x such that  (a * x) % N is 1

g, x = EEA2(a, N)
if g == 1:
    return x % N
else:
```

```
    return None
```

Example 1.8.1. Does 135 have an inverse mod 1673? If it does, find it using the Extended Euclidean Algorithm.

Solution.

1. c0, c, q, r: 1, 0, 0, 135
2. c0, c, q, r: 0, 1, 12, 53
3. c0, c, q, r: 1, -12, 2, 29
4. c0, c, q, r: -12, 25, 1, 24
5. c0, c, q, r: 25, -37, 1, 5
6. c0, c, q, r: -37, 62, 4 4
7. c0, c, q, r: 62, -285, 1, 1
8. c0, c, q, r: -285, 347, 4, 0
9. r, c: 1, 347

Therefore $135^{-1} \pmod{1673}$ is 347. And we check:

$$135 \cdot 347 = 34845 = 1 + 28 \cdot 1673 \equiv 1 \pmod{1673}$$

Exercise 1.8.2. Compute the $\gcd(16, 123)$. If it's 1, find x such that $16x \equiv 1 \pmod{123}$. Use the above version of Extended Euclidean Algorithm and compute by hand. When you're done, write a program implementing the above algorithm and check that it gives you the same result. Solve the equation

$$16x + 5 \equiv 0 \pmod{123}$$

i.e. find an integer x such that $0 \leq x < 123$ satisfying the above congruence.

Exercise 1.8.3. In your `Zmod.py`, complete the following methods:

- multiplicative inverse mod N (i.e., `inv`)
- invertibility mod N (i.e., `is_invertible`)
- division (i.e., `__div__`)

1.9 Euler Totient Function

Definition 1.9.1. Let N be a positive integer. $\phi(N)$ is the number of positive integers from 0 to $N-1$ which are coprime to N, i.e.

$$\phi(N) = |\{a \,|\, 0 \leq a \leq N-1,\ \gcd(a,N) = 1\}|$$

Note that you can also view ϕ in this way:

$$\begin{aligned}\phi(N) &= |\{a \mid 0 \leq a \leq N-1,\ \gcd(a,N) = 1\}| \\ &= |\{a \mid 0 \leq a \leq N-1,\ a \text{ is invertible mod } N\}| \\ &= |(\mathbb{Z}/N^\times)|\end{aligned}$$

Recall that $\{a \,|\, 0 \leq a \leq N-1,\ \gcd(a,N) = 1\}$ is the set of units of $\mathbb{Z}/N$ which is denoted by $(\mathbb{Z}/N)^\times$ or $U(\mathbb{Z}/N)$. So the above definition is the same as

$$\phi(N) = |U(\mathbb{Z}/N)|$$

Note that by definition $\phi(1) = 1$. Here are some important properties of ϕ.

Proposition 1.9.1. *Let $n > 0$ be a positive integer.*

(a) *Then*

$$\phi(n) = n \prod_{p|N} \left(1 - \frac{1}{p}\right)$$

where "$\prod_{p|n}$" is "product over all primes p dividing n".

(b) *If m, n are coprime, i.e. $\gcd(m,n) = 1$, then $\phi(mn) = \phi(m)\phi(n)$.*

(c) *If p is a prime and $k > 0$, then $\phi(p^k) = p^{k-1}(p-1) = p^k - p^{k-1}$.*

Proof. TODO □

Let's compute $\phi(10)$. Note that $10 = 2 \cdot 5$ and $\gcd(2,5) = 1$. Therefore using (b) of the above theorem we get

$$\phi(10) = \phi(2^1 \cdot 5^1) = \phi(2^1) \cdot \phi(5^1)$$

since $\gcd(2^1 \cdot 5^1) = 1$. Using (c) of the above theorem I get

$$\phi(10) = \phi(2^1) \cdot \phi(5^1) = (2^1 - 2^{1-1}) \cdot (5^1 - 5^{1-1}) = 1 \cdot 4 = 4$$

Of course you can also use (a) above to get

$$\phi(10) = 10 \cdot (1 - 1/2) \cdot (1 - 1/5) = 10 \cdot \frac{1}{2} \cdot \frac{4}{5} = 4$$

In both cases, we get 4. This means in $\mathbb{Z}/10$, there are 4 elements which are invertible. Running a through $\{0, 1, 2, ..., 9\}$, you'll see that invertible a's are $1, 3, 7, 9$ with inverses $1, 7, 3, 9$ respectively.

Exercise 1.9.1.

(a) Compute $\phi(735)$.
(b) Compute $\phi(900)$.
(c) Compute $\phi(263891)$.

Exercise 1.9.2. (a) Let p be a prime. What is $\phi(2p)$ in terms of p?
(b) How many solutions are there to $\phi(n) = 2n$?
(c) How many solutions are there to $\phi(n) = n/2$?

Exercise 1.9.3. Easy: What is $\phi(pq)$ as an integer expression involving p and q? Can you write it as an expression involving the sum and product of p and q? (i.e., besides constants and operators, your expression contains only $p + q$ and pq).

Exercise 1.9.4.

(a) Solve $\phi(n) = 2$, i.e., find all positive integers n such that $\phi(n) = 2$. (Hint: Write down the prime factorization of $n = p_1^{e_1} \cdots p_g^{e_g}$ and use the equation $\phi(n) = 2$.)
(b) Solve $\phi(n) = 3$.
(b) Solve $\phi(n) = 6$.

Exercise 1.9.5. Prove that

$$\phi(mn) = \phi(m)\phi(n) \cdot \frac{g}{\phi(g)}$$

where $g = \gcd(m, n)$. (Note that the above does *not* assume m, n are coprime. What is the above if m, n are coprime?)

Exercise 1.9.6. * Plot a function of the graph $y = \phi(x)$ for integer values of x running through 1 to 10000. See any pattern? Note that if you want an approximation (for instance in the asymptotics) that's not too difficult. Note the following: There are two ways to compute $\phi(n)$:

(a) $\phi(n) = |\{x \mid 0 \leq x < n-1, \ \gcd(x,n) = 1\}|$ which required Euclidean algorithm in a loop.
(b) $\phi(n) = \phi(p_1^{\alpha_1} \cdots p_g^{\alpha_g}) = (p_1^{\alpha_1} - p_1^{\alpha_1 - 1}) \cdots (p_1^{\alpha_g} - p_1^{\alpha_g - 1})$ which requires prime factorization.

The first method is slow: you need to loop your x and for each x you need to execute the EEA which has a loop. The second method is fast only if you can find the prime factorization of n. Is it possible to find $\phi(n)$ as a formula in n without finding the prime factorization of n? For instance the factorial function $n!$ has this interpolation: If

$$\Gamma(x) = \int_0^\infty t^{x-1} e^{-t} \, dt$$

then $\Gamma(n) = n!$ for positive integers n. Is there a fast interpolation of $\phi(n)$?

Exercise 1.9.7. * Can you find an n such that $\phi(n)$ divides $n + 1$?

Exercise 1.9.8. *

(a) If p is a prime, then $\phi(p) = p - 1$. Prove that if $\phi(n) = n - 1$, then n is a prime.
(b) Can you find an $n > 1$ which is not a prime (i.e. composite) and such that $\phi(n)$ divides $n - 1$? If you can find one, let me know ASAP. Or if you can prove that such as n does not exist, let me know ASAP.

Exercise 1.9.9. *

(a) Can you find some n such that

$$\phi(\phi(n)) = 1$$

(b) Write $\phi^2(n) = \phi(\phi(n))$. Can you find *all* n such that $\phi^2(n) = 1$.
(c) Write ϕ^k to the composition of k Euler ϕ. What about $\phi^3(n) = 1$? Can you find some n satisfying the above equation?

(d) What about $\phi^k(2^n) = 1$? What is the smallest k for $\phi^k(2^n) = 1$?
(e) What about $\phi^k(2^m \cdot 3^n) = 1$? What is the smallest k such that $\phi^k(2^m \cdot 3^n) = 1$?

As an aside, note that ϕ as a function has domain of $\mathbb{N}$. In this case, we say that ϕ is an **arithmetic function**. Furthermore, ϕ satisfies the property that if $\gcd(m, n) = 1$, then $\phi(mn) = \phi(m)\phi(n)$. A function $\mathbb{N} \to \mathbb{C}$ satisfying this property is said to be **multiplicative** . The Euler ϕ function is one of many multiplicative arithmetic functions. Multiplicative functions are extremely important in number theory.

arithmetic function

multiplicative

The following theorem allows you to compute powers in mod p extremely fast:

Theorem 1.9.1. (**Fermat's Little Theorem**). *Let p is a prime number and a be a positive integer not divisible by p. Then*

Fermat's Little Theorem

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

Proof. TODO □

Exercise 1.9.10. Compute r where r is the smallest positive integer satisfying

$$5^{642} \equiv r \pmod{641}$$

Corollary 1.9.1. *Let p be a prime. Then $a^p \equiv a \pmod{p}$.*

Note that the corollary does not require $p \nmid a$. The proof of the corollary is easy. If $p \mid a$, then both sides of the equation is 0 mod p, so the congruence is true. If $p \nmid a$, then Fermat's Little Theorem gives us

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

on multiplying both sides by a, we get

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

Exercise 1.9.11. What is the remainder of $3^{122436481}$ mod 13?

Note that Fermat's Little Theorem can be used to compute powers very rapidly if you work in mod p where p is a prime. What if you need to work in mod N where N is not a prime? There is a generalization of Fermat's Little Theorem due to Euler. Note that since $p - 1 = \phi(p)$, Fermat's Little Theorem

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

can be stated as

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

This statement actually holds if p is replaced by any positive integer.

Euler's Theorem

Theorem 1.9.2. (**Euler's Theorem**). *Let a and n be positive integers such that $\gcd(a, n) = 1$. Then*

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

Proof. TODO □

Note that for $\mathbb{Z}/N$, a computation of

$$a^k \pmod{N}$$

Euler's Theorem can lower the k if $\gcd(a, N)$. But after you have lowered the k, say to ℓ, you still need to compute $a^\ell \pmod{N}$. See the squaring algorithm in the RSA chapter.

Exercise 1.9.12. Compute r where r is the smallest positive integer satisfying

$$5^{642} \equiv r \pmod{640}$$

Exercise 1.9.13. What is the remainder of $3^{123456789}$ mod 100?

Exercise 1.9.14. What is the hundreds digit of $3^{123456789}$?

Exercise 1.9.15. In your `Zmod.py` complete the following:

- Exponentiation (i.e., `__pow__`)

Note that $x^{-1000} \pmod N$ is $(x^{-1})^{1000} \pmod N$. You want to first check if you can use Euler's Theorem. If it is, use the theorem to lower the exponent to say ℓ. Then use the obvious loop to compute x^ℓ and apply $\pmod N$ as frequently as possible. After you are done with the above, you might want to improve your `__pow__` by *not* use Euler's theorem if the exponent is "small". You can determine for yourself what if "small". For instance you can choose to use Euler's Theorem only when the exponent is greater than 10. (Later in the RSA chapter, we will talk about the squaring method.)

Exercise 1.9.16. Leetcode 372.
https://leetcode.com/problems/super-pow/
Your task is to calculate $a^b \pmod{1337}$ where a is a positive integer and b is an extremely large positive integer given in the form of an array. For instance for $a = 2, b = [1, 0]$, the output is 1024.

Exercise 1.9.17. Leetcode 1015.
https://leetcode.com/problems/smallest-integer-divisible-by-k/
Given a positive integer k, you need to find the length of the smallest positive integer n such that n is divisible by k, and n only contains the digit 1. Return the length of n. If there is no such n, return -1. Note: n may not fit in a 64-bit signed integer.

Exercise 1.9.18. Leetcode 1622.
https://leetcode.com/problems/fancy-sequence/
Write an API that generates fancy sequences using the append, addAll, and multAll operations. Implement the Fancy class:

- `Fancy()` Initializes the object with an empty sequence.
- `void append(val)` Appends an integer val to the end of the sequence.
- `void addAll(inc)` Increments all existing values in the sequence by an integer inc.
- `void multAll(m)` Multiplies all existing values in the sequence by an integer `m`.
- `int getIndex(idx)` Gets the current value at index `idx` (0-indexed) of the sequence modulo 109 + 7. If the index is greater or equal than the length of the sequence, return `-1`.

Exercise 1.9.19. Leetcode 1952

https://leetcode.com/problems/three-divisors/
Given an integer `n`, return `true` if `n` has exactly three positive divisors. Otherwise, return `false`. An integer `m` is a divisor of `n` if there exists an integer `k` such that `n = k * m`.

Exercise 1.9.20. https://acm.timus.ru/problem.aspx?space=1&num=1673
At the end of the previous semester the students of the Department of Mathematics and Mechanics of the Yekaterinozavodsk State University had to take an exam in network technologies. N professors discussed the curriculum and decided that there would be exactly N^2 labs, the first professor would hold labs with numbers $1, N+1, 2N+1, ..., N^2-N+1$, the second one — labs with numbers $2, N+2, 2N+2, ..., N^2-N+2$, etc. N-th professor would hold labs with numbers $N, 2N, 3N, ..., N^2$. The professors remembered that during the last years lazy students didn't attend labs and as a result got bad marks at the exam. So they decided that a student would be admitted to the exam only if he would attend at least one lab of each professor. N roommates didn't know the number of labs and professors in this semester. These students had different diligence: the first student attended all labs, the second one — only labs which numbers were a multiple of two, the third one — only labs which numbers were a multiple of three, etc... At the end of the semester it turned out that only K of these students were admitted to the exam. Find the minimal N which makes that possible.

Input: An integer K ($1 \leq K \leq 2 \cdot 10^9$).

Output: Output the minimal possible N which satisfies the problem statement. If there is no N for which exactly K students would be admitted to the exam, output 0.

Example: Input:8, output:15. Input:3, output:0.

Chapter 2

Classical ciphers

2.1 Shift cipher

Definition 2.1.1. The **shift cipher** (E, D) is given by

$$E(k, x) = x + k \pmod{26}$$

and

$$D(k, x) = x - k \pmod{26}$$

Historically the shift cipher with key $k = 3$ was used by Julius Caesar and is called the **Caesar cipher**.

2.2 Affine cipher

2.3 Vigenère cipher

2.4 Substitution cipher

2.5 Permutation cipher

2.6 Hill cipher

2.7 One-time pad cipher

2.8 Linear feedback shift register

Chapter 3

Group theory

3.1 Definitions

The most basic mathematical object is $\mathbb{Z}$. $\mathbb{Z}$ has two operations: addition and multiplication. We first abstract the study of $\mathbb{Z}$ by focusing on just one operation, the $+$.

Definition 3.1.1. $(G, *, e)$ is a **group** if G is a set and $*$ satisfies group

(C) If $x, y \in G$, then $x * y \in G$. In other words $* : G \times G \to G$ is a binary operator.
(A) If $x, y, z \in G$, then $(x * y) * z = x * (y * z)$.
(I) If $x \in G$, then there is some $y \in G$ such that $x * y = e = y * x$. y is called an **inverse** of x. Later we will see that the inverse of x is uniquely determined by x. inverse
(N) If $x \in G$, then $x * e = x = e * x$.

Definition 3.1.2. $(G, *, e)$ is an **abelian group** if $(G, *, e)$ is a group such that if $x, y \in G$, then $x * y = y * x$. In other words, $(G, *, e)$ is an abelian group if $(G, *, e)$ is group and $*$ is a commutative operator. abelian group

The reason for now including the commutativity condition in the definition for groups is because there are many important groups which are not abelian.

Chapter 4

Ring theory

Chapter 5

Field theory

Index

Bibliography

[1] Leslie Lamport, *LaTeX: a document preparation system*, Addison Wesley, Massachusetts, 2nd edition, 1994. (EXAMPLE)