Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

multiplying two β-bit integers by the ordinary method uses $\Theta(\beta^2)$ operation.
divide a β-bit integer by a shorter integer or take the remainder of a β-bit integer when divided by a shorter integer in time $\Theta(\beta^2)$ by simple algorithms though faster algorithms are known.

Z = the set of negative to positive integers
N = the set of non-negative natural numbers.

d | a (read "d divides a") means that a = kd for some integer k.
Every integer divides 0.
If a>0 and d | a (i.e. a/d=k) then |d| <= |a| since k is an integer.
If d | a, then we also say that a is a multiple of d since k is an integer.
If d does not divide a, we write $d \nmid a$.
If d | a, and d >= 0, we say that d is a divisor of a.
Note that d | a if and only if -d | a, so that no generality is lost by defining the divisors to be nonnegative, with the understanding that the negative of any divisor of a also divides a.
A divisor of a nonzero integer a is at least 1 but not greater than |a|.
For example, the divisors of 24 are 1, 2, 3, 4, 6, 8, 12, and 24.
Every positive integer a is divisible by the trivial divisors 1 and a. The nontrivial divisors of a are the factors of a.
For example, the factors of 20 are 2, 4, 5, and 10.

An integer a > 1 whose only divisors are the trivial divisors 1 and a is a prime number.
An integer a > 1 that is not prime is a composite number.
Similarly, the integer 0 and all negative integers are neither prime nor composite.

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

Equivalence classes:
https://www.statisticshowto.com/equivalence-class/#:~:text=An%20equivalence%20class%20is%20the,%2C%20they're%20called%20equivalent.

An **equivalence class** is the name that we give to the subset of S which includes all elements that are equivalent to each other.
"Equivalent" is dependent on a specified relationship, called an **equivalence relation**. If there's an equivalence relation between any two elements, they're called equivalent.

Example:
If X is the set of all integers, we can define the equivalence relation ~ by saying 'a ~ b if and only if (a – b) is divisible by 9'.
Then the equivalence class of 4 is x in (x-4)/9 = an integer (NOTE: that's also said 9 | (x-4))
and those x would include -32, -23, -14, -5, 4, 13, 22, and 31 (and a whole lot more).

Relatively prime integers
Two integers a and b are relatively prime if their only common divisor is 1, that is, if gcd(a, b) = 1

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

n | a (read “n divides a”) means that a = kn where k and a are integers and n is a positive integer.

For any integer a and any positive integer n, there exist unique integers q and r such that
 0<=r<n and a = qn + r.
For any integer a and any positive integer n, the value
<u>a mod n is the remainder (or residue)</u> of the **<u>quotient a/n</u>**:
 a mod n = a - n $\lfloor a/n \rfloor$
 <u>We have that n | a if and only if a mod n = 0.</u>
If (a mod n) = (b mod n), we write a≡b(mod n) and say that a is equivalent to b, modulo n.
In other words, a≡b(mod n) if a and b have the same remainder when divided by n.
Equivalently, a≡b(mod n) if and only if n is a divisor of b-a.
We write a≡b(mod n) if a is not equivalent to b modulo n.

We can partition the integers into n equivalence classes according to their remainders modulo n.
The equivalence class modulo n containing an integer a is
 [a]_n = {a + k*n: k ∈ Z}
  e.g. [3]_7 = {3 + k*7: k ∈ Z}
   [3]_7 = {…, -11, -4, 3, 10, 17, … : k={…, -2, -1, 0, 1, 2, …}}
Using the notation defined on page 54, we can say that writing a ∈ [b]_n is the same as writing a≡b(mod n).

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

Using the notation defined on page 54, we can say that writing a ∈ [b]_n is the same as writing a≡b(mod n). The set of all such equivalence classes is
Z_n = {[a]_n : 0 <= a <= n-1}   **(eqn 31.1)**
When you see the definition Z_n = {0,1,…,n-1}} you should read it as equivalent to eon 31.1 with the understanding that 0 represents [0]_n, 1 represents [1]_n, and so on; ***each class is represented by its smallest nonnegative element***.
You should keep the underlying equivalence classes in mind, however.
For example, if we refer to -1 as a member of Z_n, we are really referring to [n-1]_n ,
since -1 ≡ n-1 (mod n) (derived from relationship (a mod n) = (b mod n), we write a≡b(mod n)).

common divisor: if d is a common divisor of a and b then it divides each of them.
  a property of common divisors: d|a and d|b implies d|(a+b) and d|(a-b).
  and d|a and d|b implies d|(ax + by) for integers x and y.

if a|b then either |a| <= |b| or b=0 which implies a|b and b|a implies a = +-b.

if a and b are any positive integers such that a | b, then (x mod b)mod a = x mod a for any x
  and x ≡ y (mod b) implies x ≡ y (mod a) for any integers x and y.

For any non-negative a and any positive b:
  gcd(a, b) = gcd(b, a mod b)

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

The groups defined by modular addition and multiplication:
We can form two finite abelian groups by using addition and multiplication modulo n, where n is a positive integer. These groups are based on the equivalence classes of the integers modulo n, defined in Section 31.1.

We can easily define addition and multiplication operations for Zn, because the equivalence class of two integers uniquely determines the equivalence class of their sum or product.
That is, if a ≡ a'(mod n) and b ≡ b' (mod n),
a+b = a'+b'(mod n) <=== **the notation means (a' mod n) + (b' mod n)**
ab ≡ a'b'(mod n) <=== **the notation means (a' mod n) * (b' mod n)**
Thus, we define addition and multiplication modulo n, denoted +_n and ·_n, by
[a]_n +_n [b]_n = [a + b]_n; (31.18) subtraction is similar.
[a]_n ·_n [b]_n = [a*b]_n :
Use the smallest nonnegative element of each equivalence class as its representative when performing computations in Zn, that is, replace x by x mod n.

additive group modulo n : (Zn, +_n).

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

We can form two finite abelian groups by using **addition and multiplication modulo n**, where n is a positive integer. These groups are *based on the equivalence classes of the integers modulo n*, defined in Section 31.1.

We can easily define addition and multiplication operations for Z_n, because the equivalence class of two integers uniquely determines the equivalence class of their sum or product. That is, if $a \equiv a'(\text{mod } n)$ and $b \equiv b' (\text{mod } n)$, then $a+b = a'+b'(\text{ mod } n)$ <=== **the notation means a' mod n + b' mod n**
$ab \equiv a'b'(\text{mod } n)$ <=== **the notation means a' mod n * b' mod n**
Thus, we define addition and multiplication modulo n, denoted $+_n$ and $\cdot_n$, by
$[a]_n +_n [b]_n = [a + b]_n$; (31.18) subtraction is similar.
$[a]_n \cdot_n [b]_n = [a*b]_n$:
Use the smallest nonnegative element of each equivalence class as its representative when performing computations in Zn, that is, replace x by x mod n.
we define the **multiplicative group modulo n** as $(Z^*_n, \cdot_n)$. The elements of this group are the set Z^*_n of elements in Z_n that are ***relatively prime to n***, so that each one has a unique inverse, modulo n:
$Z^*_n = \{[a]_n$ is a member of $Z_n : \gcd(a,n)=1\}$.
'a' are the integers between 1 and n that are relatively prime to n (ie they do not share any factors).
for $0<=a<n$, we have $a \equiv (a+kn)(\text{mod } n)$ for all integers k.
By Exercise 31.2-3, therefore, $\gcd(a,n) = 1$ implies $\gcd(a+kn, n) = 1$ for all integers k.
Since $[a]_n = \{a+kn : k$ is a member of $Z\}$, the set Z^*_n is well defined.
An example of such a group is $Z^*_{15} = \{1,2,4,7,8,11,13,14\}$ where the group operation is multiplication modulo 15. (Here we denote an element $[a]_{15}$ as a; for example, we denote $[7]_{15}$ as 7.) Figure 31.2(b) shows the group $(Z^*_{15}, \cdot_{15})$.

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

Use the smallest nonnegative element of each equivalence class as its representative when performing computations in Zn, that is, replace x by x mod n.

additive group modulo n : (Zn, +_n).

[a]_n +_ [b]_n = [a + b]_n

Fig 31.2(a) where table column header is '[a]_6' and row header is '[b]_6' and each value = ((a + k*n) % n) + ((b + k*n) % n) with n=6 and k = any number.

+_6	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	2	3	4	5	0
2	2	3	4	5	0	1
3	3	4	5	0	1	2
4	4	5	0	1	2	3
5	5	0	1	2	3	4

multiplicative group modulo n : (Z*_n, ·_n).

[a]_n ·_ [b]_n = [a * b]_n

Fig 31.2(b)
the header col and row values for a and b are from the equivalence relation for all a's in range 0<=a<n where euclid((a + k*n) % n) = 1. Note that euclid(a, n) = 1 is the same for the a's just found.
the values in the table are ((a + k*n) % n) * ((b + k*n) % n) with n=15 and k = any number.

*_15	1	2	4	7	8	11	13	14
1	1	2	4	7	8	11	13	14
2	2	4	7	8	11	13	14	1
4	4	7	8	11	13	14	1	2
7	7	8	11	13	14	1	2	4
8	…							
11	…							
13								
14								

e.g, 8 · 11 ≡ 13 (mod 15), working in Z*_15. The identity for this group is 1.
a=8;b=11;k=1;n=15;
((a + k*n) % n) * ((b + k*n) % n) =88
88 % 15 = 13.

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

The size of $\mathbb{Z}_n^*$ is denoted $\phi(n)$. This function, known as ***Euler's phi function***, satisfies the equation

$$\phi(n) = n \prod_{p\,:\,p \text{ is prime and } p \mid n} \left(1 - \frac{1}{p}\right) , \qquad (31.20)$$

The elements of p are each prime divisor of n, that is, p are the factors of n that are prime.

so that p runs over all the primes dividing n (including n itself, if n is prime).

If n is composite, then $\phi(n) < n - 1$,

A lower bound on phi for composite n:

$$\phi(n) > \frac{n}{e^\gamma \ln\ln n + \frac{3}{\ln\ln n}}$$

for $n \geq 3$, where $\gamma = 0.5772156649$

For n=15, can see there are 8 members of Z*_n,
factors of 15 are p=(3, 5) giving phi=15*(2/3)*(4/5)

---

a=1, z=1, gcd=(1,1) (gcd, x, y) = ([1, 1, 0]) a*x+n*y=1 <=
a=2, z=2, gcd=(1,1) (gcd, x, y) = ([1, -7, 1]) a*x+n*y=1 <=
a=3, z=3, gcd=(3,3) (gcd, x, y) = ([3, 1, 0]) a*x+n*y=3
a=4, z=4, gcd=(1,1) (gcd, x, y) = ([1, 4, -1]) a*x+n*y=1 <=
a=5, z=5, gcd=(5,5) (gcd, x, y) = ([5, 1, 0]) a*x+n*y=5
a=6, z=6, gcd=(3,3) (gcd, x, y) = ([3, -2, 1]) a*x+n*y=3
a=7, z=7, gcd=(1,1) (gcd, x, y) = ([1, -2, 1]) a*x+n*y=1 <=
a=8, z=8, gcd=(1,1) (gcd, x, y) = ([1, 2, -1]) a*x+n*y=1 <=
a=9, z=9, gcd=(3,3) (gcd, x, y) = ([3, 2, -1]) a*x+n*y=3
a=10, z=10, gcd=(5,5) (gcd, x, y) = ([5, -1, 1]) a*x+n*y=5
a=11, z=11, gcd=(1,1) (gcd, x, y) = ([1, -4, 3]) a*x+n*y=1 <=
a=12, z=12, gcd=(3,3) (gcd, x, y) = ([3, -1, 1]) a*x+n*y=3
a=13, z=13, gcd=(1,1) (gcd, x, y) = ([1, 7, -6]) a*x+n*y=1 <=
a=14, z=14, gcd=(1,1) (gcd, x, y) = ([1, -1, 1]) a*x+n*y=1 <=

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

Theorem 31.14 gives us an easy way to produce a subgroup of a finite group $(S, \oplus)$: choose an element a and take all elements that can be generated from a using the group operation. Specifically, define $a^{(k)}$ for $k \geq 1$ by

$$a^{(k)} = \bigoplus_{i=1}^{k} a = \underbrace{a \oplus a \oplus \cdots \oplus a}_{k} .$$

$ax \equiv 1 \pmod{n}$
which is a*x % n = 1

For example, if we take $a = 2$ in the group $\mathbb{Z}_6$, the sequence $a^{(1)}, a^{(2)}, a^{(3)}, \ldots$ is

$2, 4, 0, 2, 4, 0, 2, 4, 0, \ldots$.

In the group $\mathbb{Z}_n$, we have $a^{(k)} = ka \bmod n$, and in the group $\mathbb{Z}_n^*$, we have $a^{(k)} = a^k \bmod n$. We define the ***subgroup generated by a***, denoted $\langle a \rangle$ or $(\langle a \rangle, \oplus)$, by

$\langle a \rangle = \{a^{(k)} : k \geq 1\}$.

from additive group modulo n
we still have example
[a]_6' = (0, 1, 2, 3, 4, 5) .

Z_6 using an equivalence relation of a^(k) = k*a mod n and let a=2: the subset <a> for k=1…
(a*1) % n = 2
(a*2) % n = 4
(a*3) % n = 0
the subgroup of [a]_6 w/ a=2 is denoted <2>.
determine a subgroup using (a*k) % n:
<0>=(0)
<1>=(0,1,2,3,4,5)
<2>=(0,2,4)

For Z*_7, first determine [a]_7 from all a's in range 0<=a<n
where euclid((a + k*n) % n) = 1
[a]_7 = (1, 2, 3, 4, 5, 6) .
determine a subgroup using (math.pow(a, k)) % n:
<1> = (1)
<2> = (2,4,1) = (1,2,4)
<3> = (3,2,6,4,5,1) = (1,2,3,4,5,6)
The order of a (in the group S=[a]_7 here), denoted ord(a), is defined as the smallest positive integer k such that a^(k)=identity where a^(k) is the notation for the group operation of the example in Corollary 31.16.
(I think that just means, only the unique members of <a> to be in the subgroup). see next 2 pages of notes…)

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

The ***order*** of a (in the group S), denoted $\mathrm{ord}(a)$, is defined as the smallest positive integer t such that $a^{(t)} = e$.

2. **Identity:** There exists an element $e \in S$, called the ***identity*** of the group, such that $e \oplus a = a \oplus e = a$ for all $a \in S$.

The identity element of $(\mathbb{Z}_n, +_n)$ is 0 (that is, $[0]_n$). The (additive) inverse of an element a (that is, of $[a]_n$) is the element $-a$ (that is, $[-a]_n$ or $[n-a]_n$), since $[a]_n +_n [-a]_n = [a-a]_n = [0]_n$. ■

Proof Theorem 31.6 implies that $(\mathbb{Z}_n^*, \cdot_n)$ is closed. Associativity and commutativity can be proved for $\cdot_n$ as they were for $+_n$ in the proof of Theorem 31.12. The identity element is $[1]_n$. To show the existence of inverses, let a be an element of $\mathbb{Z}_n^*$ and let (d, x, y) be returned by EXTENDED-EUCLID(a, n). Then, $d = 1$, since $a \in \mathbb{Z}_n^*$, and

$$ax + ny = 1 \tag{31.19}$$

or, equivalently,

$$ax \equiv 1 \pmod{n} .$$

Thus, $[x]_n$ is a multiplicative inverse of $[a]_n$, modulo n. Furthermore, we claim that $[x]_n \in \mathbb{Z}_n^*$. To see why, equation (31.19) demonstrates that the smallest pos-

Introduction to Algorithms by Cormen et al (CLRS)
Chap 31, Number Theory

The ***order*** of a (in the group S), denoted $\text{ord}(a)$, is defined as the smallest positive integer t such that $a^{(t)} = e$.

for Z*_n, the identity element is [1]_n. See Theorem 31.13 proof.

In Corollary 31.16, the missing details for the example for Z*_7 are:
    generate [1]_7 using (a*k)%n for k=1 through n-1:
        [1]_7 = (1,2,3,4,5,6)
    determine a subgroup using (math.pow(a, k)) % n:
    <1> = (1,1,1,1,1,1)
Then the smallest positive integer t such that a^(t) = [1]_7 is ‘1’
    so <1> = (1)
    <2> = (2,4,1,2,4,1)
Then the smallest positive integer t such that a^(t) = [1]_7 is ‘3’
    so <2> = (1,2,4)
    etc…