

Powers of an Element

Consider powers of a modulo n :

$$\begin{array}{l} i \\ 3^i \pmod{7} \end{array} \begin{array}{cccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & \dots \\ 1 & 3 & 2 & 6 & 4 & 5 & 1 & 3 & 2 & 6 & 4 & 5 & \dots \end{array}$$

$$\begin{array}{l} i \\ 2^i \pmod{7} \end{array} \begin{array}{cccccccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & \dots \\ 1 & 2 & 4 & 1 & 2 & 4 & 1 & 2 & 4 & 1 & 2 & 4 & \dots \end{array}$$

an example.

- **Theorem:** (Euler) $\forall n \in \mathcal{Z}$, and $n > 1$, then

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

$$\forall a \in \mathcal{Z}_n^*.$$

- **Theorem:** (Fermat's little) If p is prime, then

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

$$\forall a \in \mathcal{Z}_p^*.$$

If $\text{ord}_n(g) = |\mathcal{Z}_n^*|$, then every element in \mathcal{Z}_n^* is a power-of- g , and we call g a generator of \mathcal{Z}_n^* , or a primitive root.

Above we see 3 is a generator of \mathcal{Z}_7^* , but 2 is NOT.

Proofs of these follow from results in the modular arithmetic section.

If \mathcal{Z}_n^* has a primitive root, we say the group \mathcal{Z}_n^* is cyclic.

- **Theorem:** The values of $n > 1$ for which \mathcal{Z}_n^* is cyclic are $2, 4, p^e$, and $2p^e$ for all odd primes, p , and positive integers e .
- If g is a primitive root of \mathcal{Z}_n^* and $a \in \mathcal{Z}_n^*$, then $\exists z$ such that $g^z \equiv a \pmod{n}$, we call z the “discrete logarithm” or “index” of a , modulo n , to the base g , $z = \text{ind}_{n,g}(a)$.
- **Theorem** (discrete logarithm): If $g \in \mathcal{Z}_n^*$ is a primitive root, then the equation

$$g^x \equiv g^y \pmod{n} \text{ holds} \iff x \equiv y \pmod{\phi(n)} \text{ holds}$$

Proof: Suppose $x \equiv y \pmod{\phi(n)}$, then $x = y + k\phi(n)$ for some k , so

$$\begin{aligned} g^x &\equiv g^{y+k\phi(n)} \pmod{n} \equiv g^y \cdot g^{k\phi(n)} \pmod{n} \\ &\equiv g^y \cdot 1^k \pmod{n} \equiv g^y \pmod{n} \end{aligned}$$

Suppose $g^x \equiv g^y \pmod{n}$, since the powers of g generate $\langle g \rangle$ with $\langle g \rangle = \phi(n)$, when g is primitive. We previously proved that the powers of g are periodic with period $\phi(n)$, thus

$$g^x \equiv g^y \pmod{n} \implies x \equiv y \pmod{\phi(n)}. \quad \blacksquare$$

- **Theorem:** If p is an odd prime and $e \in \mathcal{Z} \geq 1$, then $x^2 \equiv 1 \pmod{p^e}$ has only two solutions, $x = 1$, and $x = -1 \equiv p^e - 1 \pmod{p^e}$.

Proof: The above theorem says $\mathcal{Z}_{p^e}^*$ has at least one primitive root, call it g . We rewrite the equation as:

$$(g^{\text{ind}_{n,g}(x)})^2 \equiv g^{\text{ind}_{n,g}(1)} \pmod{n}$$

but $\text{ind}_{n,g}(1) = 0$, so by the previous theorem, we must have

$$2 \cdot \text{ind}_{n,g}(x) \equiv 0 \pmod{\phi(n)}$$

$d = \gcd(2, \phi(p^e)) = \gcd(2, (p-1)p^{e-1}) = 2$. Since $p-1$ is even, when p is an odd prime, also $d = 2|0$, so this has exactly 2 solutions, which, by inspection are 1 and -1. \blacksquare

- x is called a “nontrivial square root of unity modulo n ” if $x^2 \equiv 1 \pmod{n}$ has more solutions than ± 1 .

6 is a nontrivial root of unity modulo 35 as $6^2 = 36 \pmod{35} = 1$.

- **Corollary:** If \exists a nontrivial root of unity modulo n , then n is a composite.

Proof: This is just the contrapositive of the previous theorem. ■

NOTE: We will produce witnesses to the fact that n is composite using this fact in the Miller-Rabin primality testing algorithm.

- **Exponentiation via square-and-multiply**

Given a, b , and $n \in \mathcal{Z}^+$, we often need to compute

$a^b \pmod{n}$, this is modular exponentiation. We can do it in $O(b)$ multiplications, but we show how to do this more quickly, $O(\lg b)$ multiplications through the square-and-multiply algorithm.

Let $\langle b_k, b_{k-1}, \dots, b_0 \rangle$ be the binary representation of b .

MODULAR-EXPONENTIATION(a, b, n).

1. $c \leftarrow 0$
2. $d \leftarrow 0$
3. **Let** $\langle b_k, b_{k-1}, \dots, b_0 \rangle$ be the binary representation of b .
4. **for** $i \leftarrow k$ **down to** 0
5. **do** $c \leftarrow 2c$
6. $d \leftarrow (d \cdot d) \pmod{n}$
7. **if** $b_i = 1$
8. **then** $c \leftarrow c + 1$
9. $d \leftarrow (d \cdot a) \pmod{n}$
10. **return** n

- **Note:** line 9 computes a^c , and c is the representation of b starting from the most-significant bit and working down, one bit at a time. Also, all binary number have 1 as their most-significant bits.

We have seen this in the hashing discussion.

i	9	8	7	6	5	4	3	2	1	0
b_i	1	0	0	0	1	1	0	0	0	0
c	1	2	4	8	17	35	70	140	280	560
d	7	49	157	526	160	241	298	166	67	1

Figure 31.4 The results of MODULAR-EXPONENTIATION when computing $a^b \pmod{n}$, where $a = 7$, $b = 560 = \langle 1000110000 \rangle$, and $n = 561$. The values are shown after each execution of the **for** loop. The final result is 1.