Fermat Theorem

Fermat's little theorem states that if *p* is a prime number, then for any integer *a*,

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

$$20^{7-1} \equiv 1 \pmod{7}$$

$$20^{6} \equiv 1 \pmod{7}$$

$$64,000,000 \equiv 1 \pmod{7}$$

$$\frac{64,000,000 - 1}{7} \equiv 9,142,857$$

Application Of Fermat Theorem

- To Reduce the large power of some integer.
  - Assume integer X is very large and p is a prime number.
    - Since □ $a^{p-1}\ (mod\ p) = 1\ (mod\ p)$
    - Therefore □ $a^{X}\ (mod\ p) = a^{X\ (mod\ p)}(mod\ p)$
- To calculate modular multiplicative inverse of prime numbers
    - Since □ $a^{p-1}\ (mod\ p) = 1\ (mod\ p)$
    - Therefore □ $a^{p-2}\ (mod\ p) = a^{-1}(mod\ p)$

Euler's Theorem

In number theory, **Euler's theorem** (also known as the **Fermat–Euler theorem** or **Euler's totient theorem**) states that if *n* and *a* are coprime positive integers, then

$$a^{phi(n)} (mod\, n) = 1 \ (mod\, n)$$

Euler's totient function, also known as **phi-function** $\phi(n)$, counts the number of integers between 1 and n inclusive, which are coprime to n. Two numbers are coprime if their greatest common divisor equals 1 (1 is considered to be coprime to any number).

Here are values of $\phi(n)$ for the first few positive integers:

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
$\phi(n)$	1	1	2	2	4	2	6	4	6	4	10	4	12	6	8	8	16	6	18	8	12

Euler’s Totient Function Formula

$$\phi(n) = n \cdot \left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \cdots \left(1 - \frac{1}{p_k}\right)$$

$$n = {p_1}^{a_1} \cdot {p_2}^{a_2} \cdots {p_k}^{a_k}$$

p1, p2, p3 … are prime factors of n.

For Example:
phi(7) = 7 * (1-1/7) = 6
Since 7 is itself prime, therefore all positive numbers less than it will be co-prime to it.
Phi(6) = 6*(1-1/2)*(1-1/3) = 2
Numbers co-prime to 6 are 1 and 5.

Can You write a function to calculate phi(n) on your own?
Expected time complexity - O(nloglogn)

Code for Euler Totient function

```
void phi_1_to_n(int n) {
    vector<int> phi(n + 1);
    phi[0] = 0;
    phi[1] = 1;
    for (int i = 2; i <= n; i++)
        phi[i] = i;

    for (int i = 2; i <= n; i++) {
        if (phi[i] == i) {
            for (int j = i; j <= n; j += i)
                phi[j] -= phi[j] / i;
        }
    }
}
```

This is basically → phi[j]=(phi[j]-phi[j]/prime)