Last update: December 30, 2024 Translated From: e-maxx.ru

Euler's totient function

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

Properties

The following properties of Euler totient function are sufficient to calculate it for any number:

- If p is a prime number, then $\gcd(p, q) = 1$ for all $1 \le q < p$. Therefore we have:

$$\phi(p) = p - 1.$$

- If p is a prime number and $k \ge 1$, then there are exactly p^k/p numbers between 1 and p^k that are divisible by p. Which gives us:

$$\phi(p^k) = p^k - p^{k-1}.$$

- If a and b are relatively prime, then:

$$\phi(ab) = \phi(a) \cdot \phi(b).$$

  This relation is not trivial to see. It follows from the Chinese remainder theorem. The Chinese remainder theorem guarantees, that for each $0 \le x < a$ and each $0 \le y < b$, there exists a unique $0 \le z < ab$ with $z \equiv x \pmod{a}$ and $z \equiv y \pmod{b}$. It's not hard to show that z is coprime to ab if and only if x is coprime to a and y is coprime to b. Therefore the amount of integers coprime to ab is equal to product of the amounts of a and b.

- In general, for not coprime a and b, the equation

$$\phi(ab) = \phi(a) \cdot \phi(b) \cdot \frac{d}{\phi(d)}$$

  with $d = \gcd(a, b)$ holds.

Thus, using the first three properties, we can compute $\phi(n)$ through the factorization of n (decomposition of n into a product of its prime factors). If $n = {p_1}^{a_1} \cdot {p_2}^{a_2} \cdots {p_k}^{a_k}$, where p_i are prime factors of n,

$$\begin{aligned}\phi(n) &= \phi({p_1}^{a_1}) \cdot \phi({p_2}^{a_2}) \cdots \phi({p_k}^{a_k}) \\ &= \left({p_1}^{a_1} - {p_1}^{a_1-1}\right) \cdot \left({p_2}^{a_2} - {p_2}^{a_2-1}\right) \cdots \left({p_k}^{a_k} - {p_k}^{a_k-1}\right) \\ &= p_1^{a_1} \cdot \left(1 - \frac{1}{p_1}\right) \cdot p_2^{a_2} \cdot \left(1 - \frac{1}{p_2}\right) \cdots p_k^{a_k} \cdot \left(1 - \frac{1}{p_k}\right) \\ &= n \cdot \left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \cdots \left(1 - \frac{1}{p_k}\right)\end{aligned}$$

Implementation

Here is an implementation using factorization in $O(\sqrt{n})$:

```
int phi(int n) {
    int result = n;
    for (int i = 2; i * i <= n; i++) {
        if (n % i == 0) {
            while (n % i == 0)
                n /= i;
            result -= result / i;
        }
    }
    if (n > 1)
        result -= result / n;
    return result;
}
```

Euler totient function from 1 to n in $O(n \log \log n)$

If we need the totient of all numbers between 1 and n, then factorizing all n numbers is not efficient. We can use the same idea as the Sieve of Eratosthenes. It is still based on the property shown above, but instead of updating the temporary result for each prime factor for each number, we find all prime numbers and for each one update the temporary results of all numbers that are divisible by that prime number.

Since this approach is basically identical to the Sieve of Eratosthenes, the complexity will also be the same: $O(n \log \log n)$

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

Divisor sum property

This interesting property was established by Gauss:

$$\sum_{d|n} \phi(d) = n$$

Here the sum is over all positive divisors d of n.

For instance the divisors of 10 are 1, 2, 5 and 10. Hence $\phi(1) + \phi(2) + \phi(5) + \phi(10) = 1 + 1 + 4 + 4 = 10$.

Finding the totient from 1 to n using the divisor sum property

The divisor sum property also allows us to compute the totient of all numbers between 1 and n. This implementation is a little simpler than the previous implementation based on the Sieve of Eratosthenes, however also has a slightly worse complexity: $O(n \log n)$

```
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 - 1;

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

Application in Euler's theorem

The most famous and important property of Euler's totient function is expressed in **Euler's theorem**:

$$a^{\phi(m)} \equiv 1 \pmod{m} \quad \text{if } a \text{ and } m \text{ are relatively prime.}$$

In the particular case when m is prime, Euler's theorem turns into **Fermat's little theorem**:

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

Euler's theorem and Euler's totient function occur quite often in practical applications, for example both are used to compute the modular multiplicative inverse.

As immediate consequence we also get the equivalence:

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

This allows computing $x^n \bmod m$ for very big n, especially if n is the result of another computation, as it allows to compute n under a modulo.

Group Theory

$\phi(n)$ is the order of the multiplicative group mod n $(\mathbb{Z}/n\mathbb{Z})^\times$, that is the group of units (elements with multiplicative inverses). The elements with multiplicative inverses are precisely those coprime to n.

The multiplicative order of an element a mod n, denoted $\text{ord}_n(a)$, is the smallest $k > 0$ such that $a^k \equiv 1 \pmod{m}$. $\text{ord}_n(a)$ is the size of the subgroup generated by a, so by Lagrange's Theorem, the multiplicative order of any a must divide $\phi(n)$. If the multiplicative order of a is $\phi(n)$, the largest possible, then a is a primitive root and the group is cyclic by definition.

Generalization

There is a less known version of the last equivalence, that allows computing $x^n \bmod m$ efficiently for not coprime x and m. For arbitrary x, m and $n \geq \log_2 m$:

$$x^n \equiv x^{\phi(m)+[n \bmod \phi(m)]} \mod m$$

Proof:

Let $p_1, \ldots, p_t$ be common prime divisors of x and m, and k_i their exponents in m. With those we define $a = p_1^{k_1} \ldots p_t^{k_t}$, which makes $\frac{m}{a}$ coprime to x. And let k be the smallest number such that a divides x^k. Assuming $n \geq k$, we can write:

$$\begin{aligned}
x^n \bmod m &= \frac{x^k}{a} a x^{n-k} \bmod m \\
&= \frac{x^k}{a} \left(a x^{n-k} \bmod m\right) \bmod m \\
&= \frac{x^k}{a} \left(a x^{n-k} \bmod a\frac{m}{a}\right) \bmod m \\
&= \frac{x^k}{a} a \left(x^{n-k} \bmod \frac{m}{a}\right) \bmod m \\
&= x^k \left(x^{n-k} \bmod \frac{m}{a}\right) \bmod m
\end{aligned}$$

The equivalence between the third and forth line follows from the fact that $ab \bmod ac = a(b \bmod c)$. Indeed if $b = cd + r$ with $r < c$, then $ab = acd + ar$ with $ar < ac$.

Since x and $\frac{m}{a}$ are coprime, we can apply Euler's theorem and get the efficient (since k is very small; in fact $k \leq \log_2 m$) formula:

$$x^n \bmod m = x^k \left(x^{n-k \bmod \phi(\frac{m}{a})} \bmod \frac{m}{a}\right) \bmod m.$$

This formula is difficult to apply, but we can use it to analyze the behavior of $x^n \bmod m$. We can see that the sequence of powers $(x^1 \bmod m, x^2 \bmod m, x^3 \bmod m, \ldots)$ enters a cycle of length $\phi\left(\frac{m}{a}\right)$ after the first k (or less) elements. $\phi\left(\frac{m}{a}\right)$ divides $\phi(m)$ (because a and $\frac{m}{a}$ are coprime we have $\phi(a) \cdot \phi\left(\frac{m}{a}\right) = \phi(m)$), therefore we can also say that the period has length $\phi(m)$. And since $\phi(m) \geq \log_2 m \geq k$, we can conclude the desired, much simpler, formula:

$$x^n \equiv x^{\phi(m)} x^{(n-\phi(m)) \bmod \phi(m)} \bmod m \equiv x^{\phi(m)+[n \bmod \phi(m)]} \mod m.$$

Practice Problems

- SPOJ #4141 "Euler Totient Function" [Difficulty: CakeWalk]
- UVA #10179 "Irreducible Basic Fractions" [Difficulty: Easy]
- UVA #10299 "Relatives" [Difficulty: Easy]
- UVA #11327 "Enumerating Rational Numbers" [Difficulty: Medium]
- TIMUS #1673 "Admission to Exam" [Difficulty: High]
- UVA 10990 - Another New Function
- Codechef - Golu and Sweetness

- SPOJ - LCM Sum
- GYM - Simple Calculations (F)
- UVA 13132 - Laser Mirrors
- SPOJ - GCDEX
- UVA 12995 - Farey Sequence
- SPOJ - Totient in Permutation (easy)
- LOJ - Mathematically Hard
- SPOJ - Totient Extreme
- SPOJ - Playing with GCD
- SPOJ - G Force
- SPOJ - Smallest Inverse Euler Totient Function
- Codeforces - Power Tower
- Kattis - Exponial
- LeetCode - 372. Super Pow
- Codeforces - The Holmes Children
- Codeforces - Small GCD