Divisibility (2)

Number and sum of divisors

Task: compute the number of divisors $d(n)$ and the sum of divisors $\sigma(n)$ for a given positive integer n.

Number of divisors

It's easy to see that the prime factorization of a divisor d of n has to be a subset of the prime factorization of n, e.g. $6 = 2 \cdot 3$ is a divisor of $60 = 2^2 \cdot 3 \cdot 5$. So, we only need to count all different subsets of the prime factorization of n.

Usually, the number of subsets is 2^x for a set with x elements. However, this is no longer true if there are repeated elements in the set. In our case some prime factors may appear multiple times in the prime factorization of n.

If a prime factor p appears e times in the prime factorization of n, then we can use the factor p up to e times in the subset, which means we have $e + 1$ choices (p^0 up to p^e).

Therefore, if the prime factorization of n is $p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$, where p_i are distinct prime numbers, then the number of divisors is: $d(n) = (e_1 + 1) \cdot (e_2 + 1) \cdots (e_k + 1)$

Another way of thinking about it is the following:

- If there is only one distinct prime divisor ($n = p_1^{e_1}$), then there are obviously $e_1 + 1$ divisors ($1, p_1, p_1^2, \dots, p_1^{e_1}$).
- If there are two distinct prime divisors ($n = p_1^{e_1} \cdot p_2^{e_2}$), then you can arrange all divisors in a table.

	1	p_2	p_2^2	$\dots$	$p_2^{e_2}$
1	1	p_2	p_2^2	$\dots$	$p_2^{e_2}$
p_1	p_1	$p_1 \cdot p_2$	$p_1 \cdot p_2^2$	$\dots$	$p_1 \cdot p_2^{e_2}$
p_1^2	p_1^2	$p_1^2 \cdot p_2$	$p_1^2 \cdot p_2^2$	$\dots$	$p_1^2 \cdot p_2^{e_2}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
$p_1^{e_1}$	$p_1^{e_1}$	$p_1^{e_1} \cdot p_2$	$p_1^{e_1} \cdot p_2^2$	$\dots$	$p_1^{e_1} \cdot p_2^{e_2}$

So the number of divisors is trivially $(e_1 + 1) \cdot (e_2 + 1)$.

- A similar argument can be made if there are more then two distinct prime factors, leading to the above stated formula: $d(n) = (e_1 + 1) \cdot (e_2 + 1) \cdots (e_k + 1)$.

Implementation

```
long long numberOfDivisors(long long num) {
    long long total = 1;
    for (long long i = 2; i * i <= num; i++)
        if (num % i == 0) {
            int e = 0;
            while (num % i == 0) {
                e++;
                num /= i;
            }
            total *= e + 1;
        }

    if (num > 1)
        total *= 2;

    return total;
}
```

Sum of divisors

We can use the same line of reasoning as for $d(n)$:

- If there is only one distinct prime factor ($n = p_1^{e_1}$), then the sum is $\sigma(n) = 1 + p_1 + p_1^2 + \dots + p_1^{e_1} = \frac{p_1^{e_1+1} - 1}{p_1 - 1}$ (we've used the formula for finding the sum of the first n terms of a geometric progression; see proof here).
- If there are two distinct prime factors ($n = p_1^{e_1} \cdot p_2^{e_2}$), then we can make the same table as before. The only difference is that now we now want to compute the sum instead of counting the elements. It is easy to see, that the sum of each combination can be expressed as:

$$\sigma(n) = \left(1 + p_1 + p_1^2 + \dots + p_1^{e_1}\right) \cdot \left(1 + p_2 + p_2^2 + \dots + p_2^{e_2}\right) = \frac{p_1^{e_1+1} - 1}{p_1 - 1} \cdot \frac{p_2^{e_2+1} - 1}{p_2 - 1}$$

- In general, for $n = p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$ we get the formula: $\sigma(n) = \frac{p_1^{e_1+1} - 1}{p_1 - 1} \cdot \frac{p_2^{e_2+1} - 1}{p_2 - 1} \cdots \frac{p_k^{e_k+1} - 1}{p_k - 1}$

Implementation

```
long long SumOfDivisors(long long num)
{
    long long total = 1;
    for (long long i = 2; i * i <= num; i++)
        if (num % i == 0) {
            long long sum = 1, po = i;
            while (num % i == 0)
            {
                sum += po;
                po *= i;
                num /= i;
            }
            total *= sum;
        }
    if (num > 1)
        total *= (1ll + num);
    return total;
}
```

Note that in the implementation we can actually avoid using the formula for the sum of the first n terms of a geometric progression and we can just sum the numbers.

Observations

- Both $d(n)$ and $\sigma(n)$ are multiplicative functions. A function $f(x)$ is multiplicative if for coprime numbers a and b, $f(x)$ satisfies $f(a \cdot b) = f(a) \cdot f(b)$.
- The product of the divisors of n is $\mu(n) = n^{d(n)\,/\,2}$. You can see where that formula comes from by thinking of how we can form $d(n)\ /\ 2$ pairs with the divisors of n, each having the product equal to n. For example, the divisors of 84 produce the pairs $1 \cdot 84$, $2 \cdot 42$, $3 \cdot 28$, etc., and the product is $\mu(84) = 84^6 = 351298031616$. Note that the formula works well even when $d(n)$ is odd, but you have to be careful during the implementation: $\mu(25) = 25^{1.5} = 1 \cdot 5 \cdot 25$ (the divisor *in the middle* (i.e. the square root) has to be added to the product only once).

Euler's totient function

Euler's totient function, also known as the 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 is equal to 1.

Here are the values of $\phi(n)$ for the first 21 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 $\phi(n)$ 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 $p^k / p = p^{k-1}$ numbers between 1 and p^k that are divisible by p. This gives us: $\phi(p^k) = p^k - p^{k-1} = p^k \cdot (1 - \frac{1}{p})$.
- ϕ is multiplicative, i.e. if a and b are relatively prime, then $\phi(a \cdot b) = \phi(a) \cdot \phi(b)$. It might not be that trivial to understand where this relation comes from; you can see proofs here and here (different approaches, check both for more details).

Thus, we can compute $\phi(n)$ through the factorization of n. If $n = {p_1}^{a_1} \cdot {p_2}^{a_2} \cdots {p_k}^{a_k}$, where p_i are distinct prime factors of n,

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

Therefore, we can write the following code to compute $\phi(n)$. The complexity is $O(\sqrt{n})$.

Implementation

```
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;
}
```

Finding $\phi(x)$ for all x from 1 to n

If we need $\phi(x)$ for all $1 \leq x \leq n$, factorizing all n numbers is not efficient (we would get $O(n \sqrt n)$ complexity). We will use the same idea we used for the Sieve of Eratosthenes. It is still based on the properties 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 we update the temporary results of all numbers that are divisible by that prime number.

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

Implementation

```
vector<int> 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) { // phi[i] remained untouched if i is not a multiple of any number, i.e. if it is prime
            for (int j = i; j <= n; j += i)
                phi[j] -= phi[j] / i;
        }

    return phi;
}
```

Observations

- For any positive integers a and b (not necessarily coprime), the following equation holds: $\phi(ab) = \phi(a) \cdot \phi(b) \cdot \frac{d}{\phi(d)}$, where $d = \gcd(a, b)$
- $\sum_{d \mid n} \phi(d) = n$; for example, the divisors of 10 are 1, 2, 5 and 10 and $\phi(1) + \phi(2) + \phi(5) + \phi(10) = 1 + 1 + 4 + 4 = 10$
- $a \mid b \implies \phi(a) \mid \phi(b)$; for example, 4 divides 12, therefore $\phi(4) = 2$ divides $\phi(12) = 4$
- $m \mid \phi(a^m - 1)$; for example, 2 divides $\phi(5^2 - 1) = 8$
- $\phi(n)$ is even for $n \geq 3$
- Menon's identity: $\sum_{\substack{1 \leq k \leq n \\ \gcd(n,k)=1}} \gcd(k-1, n) = \phi(n) \cdot d(n$; for example, for $n = 6$, we get $\gcd(1-1, 6) + \gcd(5-1, 6) = 2 * 4$
- Due to the fact that ϕ is multiplicative, we can recall the idea of the linear sieve, and obtain $\mathcal{O}(n)$ complexity. You can read more here.

Exercises

- Sum, number (infoarena, CSES) and product (CSES) of divisors
- Euler's totient function (pbinfo, pbinfo)
- Farey sequence (kattis)
- Let's compute $d(x)$ for $1 \leq x \leq n$ in a more efficient way (CSES). Hint: $\mathcal{O}(n \log \log n)$
- $\sum_{1 \leq d \leq n} \sigma(d) = \ ?$ (CSES). Awesome educational problem for the topics discussed here, but unfortunately the implementation requires knowledge of modular inverse. You can see the solution explained here.
- https://onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&page=show_problem&problem=2421