We loop from 1 to $\sqrt{N}-1$ at line 5. Then at line 10, we handle $\sqrt{N}$ seperately.

Using Prime Factorization

It is possible to find NOD from prime factorization of N.

Suppose $N = 12$. Its prime factorization is $2^2 \times 3$. Is it possible to divide 12 with 5? No. Cause the prime factorization of 12 does not contain 5. We can divide N with primes that appear in factorization of N.

Next, can we divide 12 with 2^3? No. The power of 2 in prime factorization of 12 is 2^2. So we cannot divide 12 with 2^3, since 2^2 is not divisible by 2^3.

So, if we are trying to divide N, that has prime factorization $N = p_1^{a_1} \times p_2^{a_2} \times \ldots p_x^{a_x}$, then the divisors of N, D, will have prime factorization of form $D = p_1^{b_1} \times p_2^{b_2} \times \ldots p_x^{b_x}$, where $b_i <= a_i$.

For example, divisor of $12 = 2^2 \times 3$ will have prime factorization, $D = 2^x \times 3^y$, where $x <= 2, y <= 1$. When $x = 1$ and $y = 1$, $D = 2^1 \times 3^1 = 6$ which is a divisor of 12. If we use a different combination of $[x, y]$, then we get a different divisor. So how many different combination can we use here? We know that, $0 <= x <= 2$ and $0 <= y <= 1$. So for x we can use $\{0, 1, 2\}$ and for y we can use $\{0, 1\}$. So we can use $3 \times 2 = 6$ combinations of $\{x, y\}$. And that is our NOD.

So, $if\ N = p_1^{a_1} \times p_2^{a_2} \times \ldots p_x^{a_x},\ then\ D = p_1^{b_1} \times p_2^{b_2} \times \ldots p_x^{b_x}$. We get new divisors for using different combination for $\{b_1, b_2 \ldots b_x\}$. How many different combinations can we make? $(a_1 + 1) \times (a_2 + 1) \times \ldots (a_x + 1)$. Therefore, $\sigma_0(N) = (a_1 + 1) \times (a_2 + 1) \times \ldots (a_x + 1)$.

So basically, in order to find NOD, all we need to do is factorize N, then take each of its prime factors power, increase them by 1 and finally multiply all of them. Easy right?

Examples

$N = 12 = 2^2 \times 3$. $NOD(N) = (2 + 1) \times (1 + 1) = 6$.
$N = 15 = 3 \times 5$. $NOD(N) = (1 + 1) \times (1 + 1) = 4$.
$N = 252 = 2^2 \times 3^2 \times 7$. $NOD(N) = (2 + 1) \times (2 + 1) \times (1 + 1) = 18$.

Try out this yourself using other small values of N.

Code Using Prime Factorization

The code for NOD is not very different from the one we used for $factorize()$ in Prime Factorization of an Integer. We just need to modify it slightly to suit our needs.

```
int NOD ( int n ) {
    int sqrtn = sqrt ( n );
    int res = 1;
    for ( int i = 0; i < prime.size() && prime[i] <= sqrtn; i++ ) {
        if ( n % prime[i] == 0 ) {
            int p = 0; /*Counter for power of prime*/
            while ( n % prime[i] == 0 ) {
                n /= prime[i];
                p++;
            }
            sqrtn = sqrt ( n );
            p++;/*Increase it by one at end*/
            res *= p; /*Multiply with answer*/
        }
    }
    if ( n != 1 ) {
        res *= 2; /*Remaining prime has power p^1. So multiply with 2*/
    }
    return res;
}
```

I have highlighted the lines that are different from $factorize()$.

Reference

1. Divisor Function - https://en.wikipedia.org/wiki/Divisor_function

Posted by Mohammad Samiul Islam

Labels: Divisors, Factorization, Number Theory

1 comment: