Basic number theory in competitive programming

Gleb Evstropov

Primality testing and factorization

1. Naïve algorithm just checks all candidate divisors from 1 to x. Runs in O(p)
2. If p is not prime, it can be represented as $x \cdot y = p$, $x > 1$, $y > 1$
3. Either x or y should be less than $\sqrt{p}$
4. Consecutively try each value from 2 to $\sqrt{p}$, divide if possible
5. Check whether remaining value is > 1
6. We have found each prime and its degree – that is called factorization
7. Every divisor can be expressed as a tuple
8. Faster factorization algorithms and primality tests are out of focus

Euclidean algorithm

1. GCD stands for Greatest Common Divisor
2. First we compute GCD of $x > 0$ and $y > 0$.
3. $\gcd(x, y) = \gcd(x + y, y)$
4. The above implies that $\gcd(x, y) = \gcd(x - y, y)$
5. The above implies that $\gcd(x, y) = \gcd(x \bmod y, y)$
6. Continue the process until $x = 0$ or $y = 0$
7. This process is often referred as Euclid algorithm

Modulo operation

1. You can add, subtract and multiply modulo non-prime m
2. If m is prime division is defined for any x and y (y > 0)
3. Little Fermat's theorem claims $a^p \equiv a \ (mod\ p)$
4. Thus $a^{p-2} \equiv a^{-1} \ (mod\ p)$ so $\frac{a}{b} \equiv a \cdot b^{p-2} \ (mod\ p)$
5. To compute $a^b mod\ c$ use binary exponentiation (by squaring)
6. If $b\ mod\ 2\ =\ 1$ then $a^b mod\ c = a^{b-1} \cdot a\ mod\ c$
7. If $b\ mod\ 2\ = 0$ then $a^b mod\ c = (a^{\frac{b}{2}})^2\ mod\ c$

Euler function

1. Euler function $\phi(n)$ is the number of co-prime (x, n), where 0 < x < n
2. $\phi(1)$ is explicitly set to 1
3. $\phi(p) = p -\ 1$ and $\phi\left(p^k\right) = p^k - p^{k-1}$
4. $\phi(a \cdot b) = \phi(a) \cdot \phi(b)$ for co-prime a and b
5. Can be computed fast if factorization is known
6. $n = p_1^{d_1} p_2^{d_2}\ p_3^{d_3}\ \ldots p_k^{d_k}$
7. $\phi(n) = \phi(p_1)\phi(p_2)\ldots\ \phi(p_k) = (p_1^{d_1} - p_1^{d_1-1})\ (p_2^{d_2} - p_2^{d_2-1})\ldots(p_k^{d_k} - p_k^{d_k-1})$

Diophantine equation

1. Polynomial equations for which integer solution is sought
2. Linear equation of one variable is trivial $ax = b$
3. Linear equation of two variables $ax + by = c$
4. Let $d = \gcd(a, b)$, if c is not divisible by d there is no solution
5. Divide a, b and c by d
6. We actually need to solve $ax + by = 1$, as we can simply multiply its solution by c to get solution for $ax + by = c$
7. If a and b are co-prime (and they are after step 5) the solution always exists and can be found with Extended Euclidean algorithm

Extended Euclidean algorithm

1. Let us have an equation $ax + by = 1$
2. Without loss of generality assume $a \geq b$
3. Let x' and y' be solutions of $(a - b)x' + by' = 1$
4. Then $x = x'$ and $y = y' + x'$
5. Let $a = kb + r$, x' and y' be such that $rx' + by' = 1$
6. Then $x = x'$ and $y = y' + kx'$
7. The above algorithm works in logarithmic time

Chinese Remainder Theorem

1. Given $n = n_1 n_2 n_3 \ldots n_k$
2. For any $i \neq$ j values n_i and n_j are co-prime
3. We know $x \equiv x_i \ (mod\ n_i)$, find any valid x from 0 to n – 1
4. If x is valid solution, x + k·n is a valid solution for any integer k
5. Consider $x \equiv x_1 \ (mod\ n_1)$ and $x \equiv x_2 \ (mod\ n_2)$
6. $x = an_1 + x_1$ for first equation and $x = bn_2 + x_2$ for second
7. Thus $an_1 + x_1 = bn_2 + x_2$, i. e. $an_1 - bn_2 = x_2 - x_1$
8. The above can be solved with Extended Euclidean algorithm
9. New equation $x \equiv an_1 + x_1 \ (mod\ n_1 n_2)$ is introduced

Sieve of Eratosthenes

1. For each integer from 1 to n find out whether it is prime
2. Initially set each prime mark $p(x)$ = true except for $p(1)$ = false
3. Go from 2 to n, if $p(x)$ is true we should exclude all integers $x \cdot y \le n$
4. Works in $O(n \log n)$ time is we consider y from 2 to n / x for any integer
5. Actual complexity is $O(n \log \log n)$ as we try y's only for prime x
6. We can compute minimal divisor (non-trivial) for every n in the same way
7. Array $d(i)$ stores minimum prime divisor of i
8. Value $d(i) = i$ means i is prime
9. Assign $d(x \cdot y) = \min(d(x \cdot y), x)$ for each y from x to n / x

Wikipedia references

1. https://en.wikipedia.org/wiki/Factorization
2. https://en.wikipedia.org/wiki/Euclidean_algorithm
3. https://en.wikipedia.org/wiki/Euler_function
4. https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes
5. https://en.wikipedia.org/wiki/Chinese_remainder_theorem
6. https://en.wikipedia.org/wiki/Diophantine_equation

Bonus read

1. https://en.wikipedia.org/wiki/RSA_(cryptosystem)
2. https://en.wikipedia.org/wiki/Miller-Rabin_primality_test
3. https://en.wikipedia.org/wiki/Pollard_rho_algorithm