Reem's Math Notes – Number Theory Tricks

1. Number of digits = floor(log10(n)) + 1
2. log(a * b) = log(a) + log(b)
Use: compare large multiplications without overflow

3. gcd(a, b) = gcd(a, b - a)
4. Any even number ≥ 4 = sum of two primes
5. Any odd number ≥ 7 = sum of three primes

6. If (n - 2) is prime, then n = 2 + prime
Meaning: some odd numbers can be written using 2 primes only
Example: 15 = 2 + 13

7. If gcd(a, b) = x → then a = x·c and b = x·d where gcd(c, d) = 1
Meaning: gcd is a common factor, the rest are coprime
Use: helps simplify numbers or find lcm

8. Number of prime factors ≤ log(n)

9. LCM(a, b) = (a * b) / gcd(a, b)

10. a ≡ b mod m means (a - b) % m == 0
11. If a % m == b % m, then a ≡ b mod m
12. (a + b) % m = ((a % m) + (b % m)) % m

13. Total divisors of n = multiply (power + 1) of each prime factor
Example: n = $2^3 * 3^2$ → total divisors = (3+1)(2+1) = 12
Use: count all divisors quickly

14. phi(n) = number of integers from 1 to n that are coprime with n
Name: Euler's Totient Function

15. a^b % m using binary exponentiation (fast power)
Time: O(log b)
Use: efficient way to calculate large powers modulo m

16. If p is prime → (p - 1)! ≡ -1 mod p
Name: Wilson’s Theorem
Use: used to check if p is prime (not common)
17. To find divisors, loop only from 1 to sqrt(n)
Use: all divisors come in pairs before and after sqrt(n)
18. 10^9+7 and 998244353 are common mod values in problems
Reason: both are prime, help avoid overflow and make division/mod easier

---

1. Number of subarrays for array of length n = (n*(n+1)/2)
2. Number of subsets for array of length n = 2^n and -1 if you don't need the empty subset.
3. if we need to make a triangle, and we have just one side of it's 3 sides.. Let's this side n
   so,
   if a is odd --> we have a, so b= $(a*a)/2$ and c = $(a*a)/2+1$;
   if a is even --> we have a, so b = $(a*a)/4-1$ and c = $(a*a)/4+1$
   How can we observe it? with generating some triangles and observe what is the relation between the 3 sides.

```
bool isPowerOfTwo(ll n)
{
    if (n <= 0) return false;
    return !(n & (n - 1));
}
```