

Number Theory

By : Shivansh (CF : shiv_codegen)

Goals :

- ❖ Modular Arithmetic
- ❖ Modular Exponentiation
- ❖ Fermat Theorem
- ❖ Modular Inverse
- ❖ Binomial Coefficient nCr

Modular Arithmetic

Modular Arithmetic is arithmetic operations involving taking the modulo with some modulus. It is usually given in problem statements so that the solution doesn't overflow in case the answer is huge.

Few operations that involve modulo are :

- ✓ Addition
- ✓ Subtraction
- ✓ Multiplication
- ✓ Division
- ✓ Exponentiation

Modular Arithmetic – Addition, Subtraction and Multiplication

Modular Addition :

$$(A + B) \% M = (A \% M + B \% M) \% M$$

Modular Subtraction :

$$(A - B) \% M = (A \% M - B \% M) \% M$$

Modular Multiplication :

$$(A * B) \% M = (A \% M * B \% M) \% M$$

Modular Exponentiation

It finds $A^B \% M$ without causing the value to overflow.

It uses the same idea of DnC which we used in binary exponentiation, but also uses the property of modular multiplication.

$$(A * B) \% M = (A \% M * B \% M) \% M$$

We can also take $A = (A \% M)$ before passing A to the function.

```
int modular_expo(int a, int b, int m)
{
    if (b == 0) return 1;
    int c = modular_expo(a, b/2, m);
    if (b % 2 == 0) return (c*c) % m;
    return (c * c % m) * a % m;
}
```

Modular Inverse

Modular Inverse refers to the reciprocal of a number with some modulo.

Modular Multiplicative Inverse of an integer a is an integer x such that $a \cdot x$ is congruent to 1 modular some modulus m .

It can be computed by two ways :

- 1) Extended Euclidian Algorithm
- 2) Fermat's Little Theorem

- Just remember Fermat's Little Theorem doesn't work when M is non-prime.
- When can't we find modular inverse ?

Fermat Theorem

If p is a prime and for some integer k which is not multiple of p , then

$$k^{p-1} \equiv 1 \pmod{p}$$

Then to find modular inverse we can write \Rightarrow

$$k^{p-2} \cdot k \equiv 1 \pmod{p}$$

Hence k^{p-2} is the multiplicative inverse of k .

$$A^{-1} \% M = A^{M-2} \% M$$

Time Complexity : $O(\log_2 M)$ when M is prime.

```
int inverse(int a, int m)
{
    return
    binary_expo(a, m-2, m);
}
```

Binomial Coefficient

We know that

$$nCr = n! / (r! * (n-r)!)$$

This can be easily computed using Fermat's little theorem.