BASICS OF NUMBER THOERY

What is Number Theory?

— Study of Integers

Integer Valued functions.

Notations:

$\mathbb{N} = \{1, 2, 3, \ldots\ldots\}$; $\mathbb{Z} = \{\ldots -2, -1, 0, 1, 2, \ldots\}$

$a \mid b$ = 'a' divides 'b' ; $b = ka$
$k \in \mathbb{Z}$

BASICS OF MODULAR ARITHMETIC

$$a \equiv b \pmod{n}$$

$\hookrightarrow r_1 = r_2$

$$n \mid (a-b)$$

$$\begin{cases} a = \underbrace{q_1}_{} n + \underbrace{r_1}_{} \\ b = q_2 n + r_2 \end{cases}$$

$r_1 = r_2$

$$0 \leq r_1, r_2 < n$$

Examples:

$$5 \equiv 12 \pmod{7}$$

Both give rem 5

$$7 \mid (5 - 12)$$

$$2 \equiv 2 \pmod{5}$$

$$(-3) \equiv 2 \pmod{5}$$

$$5 \times (-1) + \boxed{2} = -3$$

$$5 \times 0 + \boxed{2} = 2$$

$$((-3) - 2) = -5$$

$$27 \equiv 0 \pmod{9}$$

$$9 \times 1001011 \equiv 9 \times 777 \pmod{9}$$

$$\boxed{102} \equiv \overbrace{712} \pmod{10}$$

$$1712$$

$$n = 7 \qquad a \equiv b \pmod{n}$$

$$[3]_7 = \{7k+3 \mid k \in \mathbb{Z}\}$$

$$\mathbb{Z} = \begin{cases} \text{all ints div. } 7 \\ \text{all ints of form } 7k+1 \\ \qquad\qquad\qquad\quad\; 7k+2 \\ \qquad\qquad\qquad\quad\;\; \vdots \\ \qquad\qquad\qquad\quad\; 7k+6 \end{cases}$$

S is divided M ($M = 10^9 + 7$).

$$102 = 10 \times 10 + \boxed{2}$$

$$S \equiv r \pmod{M} \qquad \underbrace{0 \le r < M}$$

"%" $\qquad S \% M$

---

1. $$(p+q)\%M = (\underbrace{p\%M}_{r} + \underbrace{q\%M}_{r_2})$$

$$p = Mk + r$$

$$q = Mk_2 + r_2$$

$$\{\cancel{M(k+k_2)} + (r+r_2)\}\%M = (r+r_2)\%M$$

$$(r+r_2)\%M = (r+r_2)\%M \quad \square$$

$$(p * q) \% M = ((p \% M) * (q \% M)) \% M$$

$$= ((p \% M) * q) \% M$$

$$(p - q) \% M \stackrel{?}{=} (p \% M - q \% M) \% M$$

$$|p - q| \% M$$

$p = 6, \; q = 4, \; M = 5$

$$\underbrace{|p - q| \% M}_{2} \stackrel{?}{=} \underbrace{|\overbrace{p \% M}^{1} - \overbrace{q \% M}^{4}|}_{3} \% \overset{5}{M}$$

$$(M + p \% M - q \% M) \% M$$

long long → $\sim 9 \times 10^{18} = 2^{63} - 1$ int → $2^{31} - 1$

$P \% M$ → $P = (P + a) \% M$

$P = (P * a) \% M$

https://codeforces.com/problemset/problem/616/E

Output

Print integer s — the value of the required sum $\boxed{\text{modulo } 10^9 + 7}$.

https://codeforces.com/problemset/problem/981/H

Output

Print the number of ways to select k enumerated not necessarily distinct simple paths in such a way that for each edge either it is not contained in any path, or it is contained in exactly one path, or it is contained in all k paths, and the intersection of all paths is non-empty.

As the answer can be large, print it $\boxed{\text{modulo } 998244353}$.

NOTE:

- Use long longs
- Take Modulo at each step
- Take care of negative values

Example 1:

$\prod_{i=0}^{N-1} C[i]$

```
int ans = 0;
for(int i = 0; i < N; ++i){

    ans += C[i];

}
printf("%d\n", ans % MOD);
```

```
// int ans = 0;
long long ans = 0;

for(int i = 0; i < N; ++i){

    // ans += C[i];
    ans = (ans + C[i]) % MOD;

}
printf("%d\n", ans % MOD);
```

Assume 0 <= A[i], B[i], C[i], D[i] < MOD

Example 2:

$\sum A[i] * B[i] \ldots$

```
long long ans1 = 0, ans2 = 0;
for(int i = 0; i < N; ++i){

    ans1 = (ans1 + A[i] * B[i] * C[i] * D[i]) % MOD;

    ans2 = (ans2 + A[i] - B[i] + C[i] - D[i]) % MOD;

}
printf("%d %d\n", ans1, ans2);
```

$-2 * MOD$

```
long long ans1 = 0, ans2 = 0;
for(int i = 0; i < N; ++i){

    //ans1 = (ans1 + A[i] * B[i] * C[i] * D[i]) % MOD;
    ans1 = (ans1 + (A[i] * B[i]) % MOD * (C[i] * D[i]) % MOD) % MOD;

    //ans2 = (ans2 + A[i] - B[i] + C[i] - D[i]) % MOD;
    ans2 = (ans2 + A[i] - B[i] + C[i] - D[i] + MOD * 2) % MOD;

}
printf("%d %d\n", ans1, ans2);
```

PRIME NUMBERS:

A number with only two factors, 1 and itself.

$$\left.\begin{matrix} 5, \\ 7, \ldots \end{matrix}\right\} \qquad \overline{9,\ 27,\ 30}$$

PRIMALITY TESTING

Given a number n (assume n > 1) check if n is a prime or not

1

```
bool isprime(long long n){
    for(long long i = 2; i < n; ++i){
        if(n % i == 0){
            return false;
        }
    }
    return true;
}
```

$\rightarrow O(n)$

$$1\ \underbrace{(2,\ 3,\ \ldots,\ n-1)}\ n$$

$$\boxed{\begin{matrix} n = a \cdot b \\ 1 < a \leq b < n \end{matrix}}$$

$$a \leq \sqrt{n}$$

$$\cancel{b \geq a > \sqrt{n}}$$

$$\cancel{b \cdot a > \sqrt{n} \cdot \sqrt{n} = n}$$

$$\cancel{a \cdot b > n}$$

2

$\sqrt{\ } \{10^{12}, 10^{13}\}$ $\quad i <= sqrt(n)$

```
bool isprime(long long n){
    for(long long i = 2; i * i <= n; ++i){
        if(n % i == 0){
            return false;
        }
    }
    return true;
}
```

$\rightarrow O(\sqrt{n})$

$O(\sqrt{n}) \approx \underbrace{10^6}$

$\leq 10^8$

— Brute force first 10 primes

— $q = (p[10] + 1)$; i

if q is a prime $\longrightarrow$ add i

— For i from $17 \} \; \boxed{18} \; \boxed{19} \; \boxed{20} \to \sqrt{n}$

$O(\sqrt{n} -$

$$\boxed{\underbrace{\pi(n)}_{\#\text{ primes} < n} = \boxed{O\left(\frac{n}{\log n}\right)}}$$

$O(\sqrt{n})$

$O\left(\frac{\sqrt{n}}{\log \sqrt{n}}\right)$ $\neq \sqrt{n}$

$O\left(\frac{n}{\log n}\right)$

$O\left(\frac{a}{\log a}\right)$

PNT

Lemma: Except 2 and 3, every prime p is either of the form (6k + 1) or (6k - 1)

3

```
bool isprime(long long n){
    if(n == 2 || n == 3){
        return true;
    }
    if(n % 2 == 0 || n % 3 == 0){
        return false;
    }
    for(long long i = 6; i * i <= n; i += 6){
        if(n % (i - 1) == 0 || n % (i + 1) == 0){
            return false;
        }
    }
    return true;
}
```

$(i-1)*(i-1)<=n$

$$\boxed{O(\sqrt{n})}$$

PRIMALITY TESTING (contd.)

p_i

You will be given Q (Q <= 1,000,000) numbers in the range [2, 1000000] and you need to test whether each of given numbers is a prime

Check Primality each Query

$O(\sqrt{p_i})$

$O\left(\sum_i \sqrt{p_i}\right)$

$O(Q\sqrt{P})$

$O(P\log\log P + Q)$

Initialize with Sieve

Process each Query in $O(1)$

$O(m+n) = O(\max(m,n))$

Sieve of Eratosthenes:

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

$\boxed{1}$	2	3	~~4~~	5	~~6~~	7	~~8~~	9	10
11	~~12~~	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

1	2	3	4	5	6	7	8	9	~~10~~
11	12	13	14	~~15~~	16	17	18	19	~~20~~
21	22	23	24	~~25~~	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Sieve of Eratosthenes (contd.):

i=2

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

```
bool isprime[MAXN];

void sieve(){
    FOR(i, MAXN){
        isprime[i] = true;
    }
    isprime[0] = isprime[1] = false;
    for(int i = 2; i < MAXN; ++i){
        if(isprime[i]){
            for(int j = i * 2; j < MAXN; j += i){
                isprime[j] = false;
            }
        }
    }
}
```

for (int i=0; i<MAXN; ++i)

for (i = 0; i<Q ; ++i){ cin >> P;
if(isprime[P]) → Yes
→ No}

The sieve of Eratosthenes is a popular way to benchmark computer performance.[13] The time complexity of calculating all primes below n in the random access machine model is $O(n \log \log n)$ operations, a direct consequence of the fact that the prime harmonic series asymptotically approaches $\log \log n$. It has an exponential time

FERMAT'S LITTLE THEOREM

$$\boxed{a^{p-1} \equiv 1 \pmod{p} \qquad a \in \mathbb{N},\ p \rightarrow \text{prime},\ \gcd(a,p) = 1}$$

MODULAR DIVISION

c, a, s.t. $a \nmid c$, $a, c \in \mathbb{N} \cup \{0\}$

$M \rightarrow$ prime.

$$\frac{c}{a} \,\%\, M$$

$a, b \in \mathbb{N} \cup \{0\}$ → prime.

$$\boxed{a \times b \equiv 1} \pmod{M}$$

"b is the modular multiplicative inverse of a modulo M"

$$k = \frac{c}{a} \quad k \in \mathbb{N}$$

$$c = \underline{ka} \quad ,$$

$$cb \equiv (ka)b \equiv k(ab) \equiv k \pmod{M}$$

$$a^{M-1} \equiv 1 \pmod{M}$$

$M > 1$ is a prime.

$$a \cdot \underbrace{a^{M-2}}_{b} \equiv 1 \pmod{M}$$

Examples:

$$2 * (51) \equiv 1 \pmod{101}$$

Uniqueness of modular inverse:

$$a * b_1 \equiv a * b_2 \equiv 1 \pmod{M}$$

$$\implies b_1 \equiv b_2 \pmod{M}$$

$$\underbrace{(b_1 * a)} * b_1 \equiv \underbrace{(b_2 * a)} * b_2$$

$$b_1 \equiv b_2 \pmod{M}.$$

$$M = 10^9 + 7$$

$$a^{(10^9+5)}$$

$O(\log M)$

```
for(int i=0; i<M-2; ++i){
    ans = ans * a % M;
}
```

FAST (BINARY) EXPONENTIATION

a^{30} even $\longrightarrow (a^{15})(a^{15}) \longrightarrow (a^2)^{15}$

$a^{15} \longrightarrow a \cdot a^{14}$

$a^{14} \longrightarrow a^7 \cdot a^7$

$a^7 \longrightarrow a \cdot a^6$

$a^6 \longrightarrow a^3 \cdot a^3$

$a^3 \longrightarrow a \cdot a^2$

$a^2 \longrightarrow a \cdot a$

$n \longrightarrow O(\log_2 n)$

Binary Representation

$$n \to \begin{cases} \boxed{\frac{n}{2}} & \text{if } n = 2k \\ n-1 & \text{if } n = 2k+1 \end{cases}$$

$$\lceil \log_2 n \rceil + \underbrace{b_1(n)}_{\#\text{set bits}}$$

```cpp
#define ll long long
const ll MOD = 1e9 + 7;

ll fxp(ll a, ll n){
    if(n == 0) return 1;
    if(n % 2 == 1) return a * fxp(a, n - 1) % MOD;
    return fxp(a * a % MOD, n / 2);
}
```

a^n

$a \cdot a^{n-1}$

$(a^{\frac{n}{2}})^2$

ll t = fxp(a, n/2); return t*t%MOD;

$(a^2)^{n/2}$

MODULAR INVERSE: FERMAT'S THEOREM REVISITED

https://www.codechef.com/NOV20A/problems/CHEFSSM

The expected number of operations can be represented as a fraction $\frac{P}{Q}$, where P is a non-negative integer and Q a positive integer coprime with $998,244,353$. You should calculate $P \cdot Q^{-1}$ modulo $998,244,353$, where Q^{-1} denotes the multiplicative inverse of Q modulo $998,244,353$.

$$\frac{P}{Q} + \frac{R}{T} = \frac{A}{B} \quad \text{in } \mathbb{Q}$$

$$AB^{-1} \equiv P \cdot Q^{-1} + RT^{-1} \pmod{M}$$

$$a \cdot b \equiv 1 \pmod{n}$$

$$\gcd(a, n) = 1$$

$$2 \cdot \cancel{b} \equiv 1 \pmod{14}$$

```cpp
#define ll long long
const ll MOD = 1e9 + 7;

ll fxp(ll a, ll n){
    if(n == 0) return 1;
    if(n % 2 == 1) return a * fxp(a, n - 1) % MOD;
    return fxp(a * a % MOD, n / 2);
}

ll inv(ll a){
    return fxp(a % MOD, MOD - 2);
}
```

$\to a^n$

$a^{(MOD-2)} \longrightarrow a^{-1}$

$O(\log MOD)$

$\#$ r distinct vases from n distinct vases. $\longrightarrow$ modulo $M = 10^9+7$

$1 \le n, r \le 10^5$

$$\underbrace{\binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r}}_{O(nr)}$$

$(j!) \% M$

$\underbrace{\qquad}$

$\forall j \in \{1, 2, 3, \ldots, n\}$

$O(n)$

$O(n) + O(\log MOD)$

$+ O(\log MOD)$

```
fact[0] = 1; ifact[0] = 1;
for (i = 1; i <= 10^5; ++i) {
    fact[i] = fact[i-1] * i % MOD
    ifact[i] = inv(fact[i]);
}
return fact[n]
```

$O(n)$

$MOD-2$

$(fact[n-r])^{-2}_{MOD} * (fact[r])$

{ # r distinct vases from n distinct vases. $\longrightarrow$ modulo $M = 10^9+7$ }

Q occurances of (n, r) $\quad {}^nC_r$

$1 \le n, r \le 10^5$

$1 \le Q \le 10^5$

$O(Q(n+\log MOD))$

$O(n+Q+n\log MOD)$

$O(n+Q\log MOD)$

```
C(n, r){
    return fact[n] * ifact[n-r] % MOD * ifact[r] % MOD
}
```

$O(Q+n\log MOD)$

APPLICATION IN COMBINATORICS

$$\binom{n}{r} = {}^{n}C_{r} = \frac{n!}{r!\,(n-r)!} \longrightarrow$$

\# ways to choose r distinct objects from n distinct objects

```cpp
#define ll long long
const ll MOD = 1e9 + 7;

ll fxp(ll a, ll n){
    if(n == 0) return 1;
    if(n % 2 == 1) return a * fxp(a, n - 1) % MOD;
    return fxp(a * a % MOD, n / 2);
}

ll inv(ll a){
    return fxp(a % MOD, MOD - 2);
}

const ll MAXN = 2e5 + 5;
ll fact[MAXN + 1], ifact[MAXN + 1];
void init(){
    fact[0] = ifact[0] = 1;
    FORe(i, MAXN){   for(int i=1; i<=MAXN; ++i){
        fact[i] = fact[i - 1] * i % MOD;
        ifact[i] = inv(fact[i]);
    }
}

ll C(ll n, ll r){
    return (r > n || r < 0) ? 0 : (ifact[r] * ifact[n - r] % MOD * fact[n] % MOD);
}
```

a^n

a A's, b B's, c C's $\longrightarrow \dfrac{(a+b+c)!}{a!\,b!\,c!}$

Proof of FLiT:

$p \to$ prime

$$a^{p-1} \equiv 1 \pmod{p} \quad \text{for } \gcd(a,p)=1$$

$$\boxed{a^p \equiv a \pmod{p}} \quad \text{for } \forall a \in \mathbb{N}$$

Induction on a:

Lemma: $\binom{p}{r}$ is a multiple of p, $0 < r < p$

$0 < p-r < p$

$$\binom{p}{r} = \frac{p!}{r!\,(p-r)!} = \frac{1\cdot 2\cdot 3 \cdot \ldots \cdot p}{(1\cdot 2 \cdots r)(1\cdot 2 \cdots (p-r))}$$

is

Basis: $a=2$

$$2^p = (1+1)^p = \sum_{i=0}^{p} \binom{p}{i} = \binom{p}{0} + \sum_{i=1}^{p-1} \binom{p}{i} + \binom{p}{p}$$

$$2^p \equiv \left\{ \underbrace{\binom{p}{0}}_{1} + \underbrace{\sum_{i=1}^{p-1} \binom{p}{i}}_{} + \underbrace{\binom{p}{p}}_{1} \right\} \pmod p$$

$$2^p \equiv 2 \pmod p$$

$$a^p \equiv a \pmod{p} \qquad a \geq 2$$

$$(a+1)^p \equiv \; ? \pmod{p}$$

$$(a+1)^p \equiv \sum_{i=0}^{p} \binom{p}{i} a^i = 1 \cdot \binom{p}{0} + \overbrace{\cancel{\sum_{i=1}^{p-1} \binom{p}{i} \cdot a^i}} + \binom{p}{p} \cdot a^p$$

$$(a+1)^p \equiv 1 \binom{p}{0} + \binom{p}{p} \cdot a^p \pmod{p}$$

$$\equiv 1 + a^p \equiv (1+a) \pmod{p}$$

MATRIX EXPONENTIATION

A^n, $n \in \mathbb{N}$

$$A = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix}$$

$A * A$

```
#define ll long long
const ll MOD = 1e9 + 7;

ll fxp(ll a, ll n){
    if(n == 0) return 1;
    if(n % 2 == 1) return a * fxp(a, n - 1) % MOD;
    return fxp(a * a % MOD, n / 2);
}
```

a^n

$a \cdot a^{n-1}$

$(a^{\frac{n}{2}})^2$

ll t = fxp(a, n/2); return t*t% MOD;

$(a^2)^{n/2}$

$A * fxp(A, n-1)$

```
for(i=0; i<n; ++i)
  for j
    for k
      R[i][j] += A[i][k] * B[k][j]
}
```

$R = A * B$

Fibonacci Numbers: f_i

$$\left.\begin{aligned} f_0 &= 0 \\ f_1 &= 1 \\ f_i &= f_{i-1} + f_{i-2} \ ; \quad i \geq 2 \end{aligned}\right\}$$

Q: $\underline{(f_N \% M)}$ → prime, 10^9+7 $\quad N \leq 10^5$

for i from 2 to 10^5

$f_i := (f_{i-1} + f_{i-2}) \% M$

$$Q \le 10^5, \quad N \le 10^{18}$$

$$\boxed{F_n = \begin{pmatrix} f_{n-1} \\ f_n \end{pmatrix}}$$

$$\underbrace{\begin{pmatrix} f_{n-1} \\ f_n \end{pmatrix}}_{F_n} = \begin{pmatrix} 0 \cdot f_{n-2} + 1 \cdot f_{n-1} \\ 1 \cdot f_{n-2} + 1 \cdot f_{n-1} \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}}_{A} \cdot \underbrace{\begin{pmatrix} f_{n-2} \\ f_{n-1} \end{pmatrix}}_{F_{n-1}} \rightarrow F_n = A \cdot F_{n-1}$$

$$F_2 = A \cdot F_1$$

$$A \cdot F_1$$

$$F_3 = \begin{pmatrix} f_2 \\ f_3 \end{pmatrix} = A \cdot F_2 = A^2 F_1$$

$$F_4 = A^3 F_1$$

$$F_5 = A^4 F_1$$

$$F_n = \underline{A}^{n-1} F_1$$

$$\underbrace{A^{n-1} = \begin{pmatrix} a & b \\ c & d \end{pmatrix}} \Longrightarrow O(\log n)$$

$$\overbrace{\begin{pmatrix} f_{n-1} \\ f_n \end{pmatrix}}^{F_n} = \overbrace{\begin{pmatrix} a & b \\ c & d \end{pmatrix}}^{A^{n-1}} \overbrace{\begin{pmatrix} f_0 \\ f_1 \end{pmatrix}}^{F_1}$$

$$\underbrace{\begin{pmatrix} f_{n-1} \\ f_n \end{pmatrix}} \overset{O(\log N)}{=} \underbrace{\begin{pmatrix} a & b \\ c & \boxed{d} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix}}$$

PROBLEMS:

* https://codeforces.com/problemset/problem/577/A
* https://codeforces.com/problemset/problem/1051/B
* https://codeforces.com/problemset/problem/1325/A
* https://codeforces.com/problemset/problem/1149/A
* https://codeforces.com/problemset/problem/230/B
* https://codeforces.com/problemset/problem/1658/B
* https://codeforces.com/contest/1662/problem/H