# 数论—卢卡斯定理、求组合数

## 说明

温馨提示：组合数一般较大，下面的示范代码均无视数据范围，如果爆 `int` 请自行开 `long long` 或高精度处理。

## 引入

从 $n$ 个不同元素中，任取 $m$ 个元素组成一个集合，叫做从 $n$ 个不同元素中取出 $m$ 个元素的一个组合；从 $n$ 个不同元素中取出 $m \le n$ 个元素的所有组合的个数，叫做从 $n$ 个不同元素中取出 $m$ 个元素的组合数，也被称为「二项式系数」。

用符号 $\dbinom{n}{m}$ 来表示，读作「$n$ 选 $m$」；组合数计算公式：$\dbinom{n}{m} = \dfrac{n!}{m!\,(n-m)!}$

特别地，规定当 $m > n$ 时，$\dbinom{n}{m} = 0$。

> 组合数也常用 $\mathrm{C}(n,m)$ 表示，即 $\mathrm{C}(n,m) = \dbinom{n}{m}$；
> 但现在数学界普遍采用 $\dbinom{n}{m}$ 的记号而非 $\mathrm{C}(n,m)$。

## 性质

$$\binom{n}{m} = \binom{n}{n-m} \tag{1}$$

$n$ 选 $m$，等价于从 $n$ 个中挑出 $n-m$ 个不选.

$$\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1} \tag{2}$$

第 $n$ 个是否选？若选，则 $n-1$ 选 $m-1$；若不选，则选 $m$ 个.

$$\sum_{i=0}^{m}\binom{n}{i}\binom{m}{m-i}=\binom{m+n}{m}\quad(n\geq m) \tag{3}$$

$n$ 选 $i$，另外 $m$ 选 $m-i$；即 $n+m$ 选 $m$.

$$\sum_{i=0}^{n}\binom{n}{i}^2=\binom{2n}{n} \tag{4}$$

$n=m$ 时的 $(3)$.

$$\binom{n}{r}\binom{r}{k}=\binom{n}{k}\binom{n-k}{r-k} \tag{5}$$

$n$ 选 $r$，再 $r$ 选 $k$ 个；即 $n$ 选 $k$，再有剩余的 $n-k$ 选 $r-k$.

$$\binom{n}{m} \bmod p=\binom{n \bmod p}{m \bmod p}\binom{n/p}{m/p} \bmod p \tag{6}$$

这个就是卢卡斯定理，见下面～

# 卢卡斯定理

## 定义

定义：$\displaystyle\binom{n}{m} \bmod p=\binom{n \bmod p}{m \bmod p}\binom{n/p}{m/p} \bmod p.$

其中 $p$ 为质数；且当 $p$ 较小时使用卢卡斯定理求解组合数较快.

## 代码实现

$\mathrm{Lucas}(n,m)=\mathrm{C}(n \bmod p, m \bmod p)\times \mathrm{Lucas}(n/p,m/p) \bmod p.$

有递归版和非递归版：

```
int lucas(int a, int b, const int p)
{
    if (a < p && b < p)
        return comb(a, b, p);
    return comb(a % p, b % p, p) * lucas(a / p, b / p, p) % p;
}
```

```
int lucas(int a, int b, const int p, int r = 1)
{
    while (a >= p || b >= p)
        r = r * comb(a % p, b % p, p) % p, a /= p, b /= p;
```

```
    return r * comb(a, b, p) % p;
}
```

### 拓展应用

分析公式，很显然是将 $n$ 和 $m$ 拆解为 $p$ 进制的过程：

$$n=\prod_{i=0}^{r} N_i\,p^k、m=\prod_{i=0}^{r} M_i\,p^k，那么 \binom{n}{m}=\prod_{i=1}^{r}\binom{N_i}{M_i}.$$

如题：P6669 组合数问题（将问题通过卢卡斯定理转化为范围内数码的问题，并用数位 DP 求解）

▼ 点击查看代码

```
typedef long long ll;

const int N = 110;
const ll MOD = 1e9 + 7;

inline ll MUL(ll a, ll b) { return (a % MOD) * (b % MOD) % MOD; }

int t, k, r;
int a[N], b[N];

ll dp[N][2][2];
ll dfs(int now, bool ln, bool lm)
{
    if (!now) return 1;
    if (dp[now][ln][lm] != -1) return dp[now][ln][lm];
    ll res = 0;
    int upn = ln ? a[now] : k - 1, upm = lm ? b[now] : k - 1;
    for (int i = 0; i <= upn; ++i) for (int j = 0; j <= i && j <= upm;
++j)
        res = (res + dfs(now - 1, ln && i == upn, lm && j == upm)) %
MOD;
    return dp[now][ln][lm] = res;
}

int main()
{
    t = rr, k = rr;
    while (t--)
```

```cpp
    {
        memset(dp, -1, sizeof dp);

        ll n = rr, m = min(n, rr);
        ll rt = (MUL(m + 1, m + 2) * 500000004ll % MOD + MUL(n - m, m + 1)) % MOD;

        int r = 0, ra = 0;
        while (n) a[++r] = n % k, n /= k;
        while (m) b[++ra] = m % k, m /= k;
        while (ra < r) b[++ra] = 0;

        printf("%lld\n", (rt - dfs(r, 1, 1) + MOD) % MOD);
    }
    return 0;
}
```

# 求组合数

### 递推预处理所有组合数

公式：$\binom{n}{m} = \binom{n-1}{m} + \binom{n-1}{m-1}$.

▼ 点击查看题目

网址：https://www.acwing.com/problem/content/887/

```cpp
const int N = 2010;
const long long MOD = 1e9 + 7;

long long comb[N][N];

int main()
{
    comb[0][0] = 1;
    for (int i = 1; i < N; ++i)
    {
        comb[i][0] = 1;
        for (int j = 1; j <= i; ++j)
            comb[i][j] = (comb[i - 1][j - 1] + comb[i - 1][j]) % MOD;
    }

    int t = rr, a, b;
```

```
    while (t--)
        a = rr, b = rr, printf("%lld\n", comb[a][b]);
    return 0;
}
```

### 预处理阶乘和逆元

定义式：$\binom{n}{m} = \frac{n!}{m!\,(n-m)!}$.

可以每次都用费马小定理计算逆元，更好的方法是线性预处理逆元；

因此也需要保证模数 $p > n, m$.

▼ 点击查看题目

网址：https://www.acwing.com/problem/content/888/

注意除去的是阶乘的逆元，所以不需要预处理单个数的逆元了.

```
const int N = 1e5 + 10;
const ll MOD = 1e9 + 7;

ll s[N], sv[N];
ll inv[N];

ll qpow(ll a, ll b, const ll p, ll r = 1)
{
    for (; b; b >>= 1)
        b & 1 ? r = r * a % p, a = a * a % p : a = a * a % p;
    return r;
}

int main()
{
    s[0] = 1;
    for (int i = 1; i < N; ++i)
        s[i] = s[i - 1] * i % MOD;

    sv[N - 1] = qpow(s[N - 1], MOD - 2, MOD);
    for (int i = N - 1; i >= 1; --i)
        sv[i - 1] = sv[i] * i % MOD;

    inv[0] = 1;
    for (int i = 1; i < N; ++i)
```

```
            inv[i] = sv[i] * s[i - 1] % MOD;

    int t = rr, a, b;
    while (t--)
        a = rr, b = rr, printf("%lld\n", s[a] * inv[b] % MOD * inv[a - b] % MOD);
    return 0;
}
```

### 卢卡斯定理求组合数

公式: $\binom{n}{m} \bmod p = \binom{n \bmod p}{m \bmod p}\binom{n/p}{m/p} \bmod p.$

▼ 点击查看题目

网址: https://www.acwing.com/problem/content/889/

```
ll qpow(ll a, ll b, const ll p, ll r = 1)
{
    for (; b; b >>= 1)
        b & 1 ? r = r * a % p, a = a *a % p : a = a * a % p;
    return r;
}

ll comb(ll a, ll b, const ll p, ll r = 1)
{
    for (int i = a, j = 1; j <= b; --i, ++j)
        r = r * i % p * qpow(j, p - 2, p) % p;
    return r;
}

int lucas(ll a, ll b, const ll p, ll r = 1)
{
    while (a >= p || b >= p)
        r = r * comb(a % p, b % p, p) % p, a /= p, b /= p;
    return r * comb(a, b, p) % p;
}

int main()
{
    ll t = rr, a, b, p;
    while (t--)
        a = rr, b = rr, p = rr, printf("%lld\n", lucas(a, b, p));
```

```
    return 0;
}
```

▼ 点击查看题目

题目：https://www.luogu.com.cn/problem/P3807

```
// 这里同上...
int main()
{
    ll t = rr, a, b, p;
    while (t--)
        a = rr, b = rr, p = rr, printf("%lld\n", lucas(a + b, a, p));
    return 0;
}
```

# Reference


[1] https://oi-wiki.org/math/number-theory/lucas/

[2] https://oi-wiki.org/math/combinatorics/combination/

[3] https://www.acwing.com/blog/content/406/




---

合集：学习笔记

标签：学习笔记 ，算法