组合数学

组合数

debug

提供一组测试数据：$\binom{132}{66} = $ 377'389'666'165'540'953'244'592'352'291'892'721'700，模数为 998244353 时为 $241'200'029$；10^9+7 时为 598375978。

逆元+卢卡斯定理（质数取模）

$\mathcal{O}(N)$，模数必须为质数。

```
struct Comb {
    int n;
    vector<Z> _fac, _inv;

    Comb() : _fac{1}, _inv{0} {}
    Comb(int n) : Comb() {
        init(n);
    }
    void init(int m) {
        if (m <= n) return;
        _fac.resize(m + 1);
        _inv.resize(m + 1);
        for (int i = n + 1; i <= m; i++) {
            _fac[i] = _fac[i - 1] * i;
        }
        _inv[m] = _fac[m].inv();
        for (int i = m; i > n; i--) {
            _inv[i - 1] = _inv[i] * i;
        }
        n = m;
    }
    Z fac(int x) {
        if (x > n) init(x);
        return _fac[x];
    }
    Z inv(int x) {
        if (x > n) init(x);
        return _inv[x];
    }
    Z C(int x, int y) {
        if (x < 0 || y < 0 || x < y) return 0;
        return fac(x) * inv(y) * inv(x - y);
    }
    Z P(int x, int y) {
        if (x < 0 || y < 0 || x < y) return 0;
        return fac(x) * inv(x - y);
    }
} comb(1 << 21);
```

质因数分解

此法适用于：$1 < n, m, MOD < 10^7$ 的情况。

```
int n,m,p,b[10000005],prime[1000005],t,min_prime[10000005];
void euler_Prime(int n){//用欧拉筛求出1~n中每个数的最小质因数的编号是多少，保存在min_prime中
    for(int i=2;i<=n;i++){
        if(b[i]==0){
            prime[++t]=i;
            min_prime[i]=t;
        }
        for(int j=1;j<=t&&i*prime[j]<=n;j++){
            b[prime[j]*i]=1;
            min_prime[prime[j]*i]=j;
            if(i%prime[j]==0) break;
        }
    }
}
long long c(int n,int m,int p){//计算C(n,m)%p的值
    euler_Prime(n);
    int a[t+5];//t代表1~n中质数的个数 ，a[i]代表编号为i的质数在答案中出现的次数
    for(int i=1;i<=t;i++) a[i]=0;//注意清0，一开始是随机数
    for(int i=n;i>=n-m+1;i--){//处理分子
        int x=i;
        while (x!=1){
            a[min_prime[x]]++;//注意min_prime中保存的是这个数的最小质因数的编号（1~t）
            x/=prime[min_prime[x]];
        }
    }
    for(int i=1;i<=m;i++){//处理分母
        int x=i;
        while (x!=1){
            a[min_prime[x]]--;
            x/=prime[min_prime[x]];
        }
    }
    long long ans=1;
    for(int i=1;i<=t;i++){//枚举质数的编号，看它出现了几次
        while(a[i]>0){
            ans=ans*prime[i]%p;
            a[i]--;
        }
    }
    return ans;
}
int main(){
    cin>>n>>m;
    m=min(m,n-m);//小优化
    cout<<c(n,m,MOD);
}
```

杨辉三角（精确计算）

60 以内 `long long` 可解，130 以内 `__int128` 可解。

```
vector C(n + 1, vector<int>(n + 1));
C[0][0] = 1;
for (int i = 1; i <= n; i++) {
    C[i][0] = 1;
    for (int j = 1; j <= n; j++) {
        C[i][j] = C[i - 1][j] + C[i - 1][j - 1];
    }
}
cout << C[n][m] << endl;
```

常见结论

n个球 全部放入 m个盒子

I：球互不相同，盒子互不相同。

每个球都有m种选择，根据乘法原理，答案是m^n

II：球互不相同，盒子互不相同，每个盒子至多装一个球。

$n > m$, 放不下，0 种可能
$n <= m$, $A(m, n)$

III：球互不相同，盒子互不相同，每个盒子至少装一个球。

容斥枚举空盒个数
$\sum_{i=0}^{m}(-1)^i * C(m, i) * (m-i)^n$

IV：球互不相同，盒子全部相同。

枚举几个盒子有球，第二类斯特林数，设f[k][b]为将k个互异元素分为b个不为空的集合
$ans = \sum_{i=0}^{m} f[n][i]$

V：球互不相同，盒子全部相同，每个盒子至多装一个球。

$n > m, 0$
$n <= m, 1$

VI：球互不相同，盒子全部相同，每个盒子至少装一个球。

正是第二类斯特林数的定义，答案是 $f[n][m]$

VII：球全部相同，盒子互不相同。

插板法，$C(n+m-1, m-1)$

VIII：球全部相同，盒子互不相同，每个盒子至多装一个球。

选n个盒子放球，$C(m,n)$

IX：球全部相同，盒子互不相同，每个盒子至少装一个球。

先把每个盒子放一个球，然后转化为第VII个问题，插板法
$C(n-1,m-1)$

X：球全部相同，盒子全部相同。

问题等价于将 $n+m$ 拆分为 m 个无序的正整数，根据 $Ferrers$ 图的理论，
等价于将 $n+m$ 拆分成若干个不超过 m 的正整数，直接生成函数做。
$T(n,m)=T(n,m-1)+T(n-m,m)$
$[x^{n+m-m}]\prod_{i=1}^{m}\frac{1}{1-x^i}$

XI：球全部相同，盒子全部相同，每个盒子至多装一个球。

同 V

XII：球全部相同，盒子全部相同，每个盒子至少装一个球。

$[x^{n-m}]\prod_{i=1}^{m}\frac{1}{1-x^i}$

```
#include <algorithm>
#include <cstdio>
int n, m, fac[400010], minv[400010];
int const mod = 998244353, g = 3, gi = (mod + 1) / g;
int C(int x, int y)
{
    if (x < 0 || y < 0 || x < y)
        return 0;
    else
        return 1ll * fac[x] * minv[y] % mod * minv[x - y] % mod;
}
int pow(int x, int y)
{
    int res = 1;
    while (y) {
        if (y & 1)
            res = 1ll * res * x % mod;
        x = 1ll * x * x % mod;
        y >>= 1;
    }
    return res;
}
struct NTT {
    int r[800010], lim;
    NTT()
        : r()
        , lim()
    {
    }
    void getr(int lm)
    {
```

```
        lim = lm;
        for (int i = 0; i < lim; i++)
            r[i] = (r[i >> 1] >> 1) | ((i & 1) * (lim >> 1));
    }
    void operator()(int* a, int type)
    {
        for (int i = 0; i < lim; i++)
            if (i < r[i])
                std::swap(a[i], a[r[i]]);
        for (int mid = 1; mid < lim; mid <<= 1) {
            int rt = pow(type == 1 ? g : gi, (mod - 1) / (mid << 1));
            for (int j = 0, r = mid << 1; j < lim; j += r) {
                int p = 1;
                for (int k = 0; k < mid; k++, p = 1ll * p * rt % mod) {
                    int x = a[j + k], y = 1ll * a[j + mid + k] * p % mod;
                    a[j + k] = (x + y) % mod, a[j + mid + k] = (x - y + mod) % mod;
                }
            }
        }
        if (type == -1)
            for (int i = 0, p = pow(lim, mod - 2); i < lim; i++)
                a[i] = 1ll * a[i] * p % mod;
    }
} ntt;
void inv(int const* a, int* ans, int n)
{
    static int tmp[800010];
    for (int i = 0; i < n << 1; i++)
        tmp[i] = ans[i] = 0;
    ans[0] = pow(a[0], mod - 2);
    for (int m = 2; m <= n; m <<= 1) {
        int lim = m << 1;
        ntt.getr(lim);
        for (int i = 0; i < m; i++)
            tmp[i] = a[i];
        ntt(tmp, 1), ntt(ans, 1);
        for (int i = 0; i < lim; i++)
            ans[i] = ans[i] * (2 - 1ll * ans[i] * tmp[i] % mod + mod) % mod, tmp[i] = 0;
        ntt(ans, -1);
        for (int i = m; i < lim; i++)
            ans[i] = 0;
    }
}
void inte(int const* a, int* ans, int n)
{
    for (int i = n - 1; i; i--)
        ans[i] = 1ll * a[i - 1] * pow(i, mod - 2) % mod;
    ans[0] = 0;
}
void der(int const* a, int* ans, int n)
{
    for (int i = 1; i < n; i++)
```

```
        ans[i - 1] = 1ll * i * a[i] % mod;
    ans[n - 1] = 0;
}
void ln(int const* a, int* ans, int n)
{
    static int b[800010];
    for (int i = 0; i < n << 1; i++)
        ans[i] = b[i] = 0;
    inv(a, ans, n);
    der(a, b, n);
    int lim = n << 1;
    ntt.getr(lim);
    ntt(b, 1), ntt(ans, 1);
    for (int i = 0; i < lim; i++)
        b[i] = 1ll * ans[i] * b[i] % mod, ans[i] = 0;
    ntt(b, -1);
    for (int i = n; i < lim; i++)
        b[i] = 0;
    inte(b, ans, n);
}
void exp(int const* a, int* ans, int n)
{
    static int f[800010];
    for (int i = 0; i < n << 1; i++)
        ans[i] = f[i] = 0;
    ans[0] = 1;
    for (int m = 2; m <= n; m <<= 1) {
        int lim = m << 1;
        ln(ans, f, m);
        f[0] = (a[0] + 1 - f[0] + mod) % mod;
        for (int i = 1; i < m; i++)
            f[i] = (a[i] - f[i] + mod) % mod;
        ntt.getr(lim);
        ntt(f, 1), ntt(ans, 1);
        for (int i = 0; i < lim; i++)
            ans[i] = 1ll * ans[i] * f[i] % mod, f[i] = 0;
        ntt(ans, -1);
        for (int i = m; i < lim; i++)
            ans[i] = 0;
    }
}
void solve1() { printf("%d\n", pow(m, n)); }
void solve2()
{
    if (m < n)
        puts("0");
    else
        printf("%lld\n", 1ll * fac[m] * minv[m - n] % mod);
}
void solve3()
{
    if (n < m)
        return puts("0"), void();
```

```
    int ans = 0;
    for (int i = 0; i <= m; i++)
        ans = (ans + 1ll * pow(mod - 1, i) * C(m, i) % mod * pow(m - i, n)) % mod;
    printf("%d\n", ans);
}
int s[800010];
void solve4()
{
    static int tmp[800010];
    for (int i = 0; i <= n; i++)
        tmp[i] = (i & 1 ? mod - 1ll : 1ll) * minv[i] % mod, s[i] = 1ll * pow(i, n) * minv[i] % mod;
    int lim = 1;
    for (lim = 1; lim <= n + n; lim <<= 1)
        ;
    ntt.getr(lim);
    ntt(tmp, 1), ntt(s, 1);
    for (int i = 0; i < lim; i++)
        s[i] = 1ll * s[i] * tmp[i] % mod;
    ntt(s, -1);
    for (int i = n + 1; i < lim; i++)
        s[i] = 0;
    int ans = 0;
    for (int i = 0; i <= m; i++)
        ans = (ans + s[i]) % mod;
    printf("%d\n", ans);
}
void solve5() { printf("%d\n", int(m >= n)); }
void solve6() { printf("%d\n", s[m]); }
void solve7() { printf("%d\n", C(n + m - 1, m - 1)); }
void solve8() { printf("%d\n", C(m, n)); }
void solve9() { printf("%d\n", C(n - 1, m - 1)); }
int ans[800010];
void solve10()
{
    static int tmp[800010];
    for (int i = 1; i <= m; i++)
        for (int j = 1; j * i <= n; j++)
            ans[i * j] = (ans[i * j] - 1ll * minv[j] * fac[j - 1] % mod + mod) % mod;
    int lim = 1;
    for (; lim <= n; lim <<= 1)
        ;

    exp(ans, tmp, lim);
    for (int i = 0; i < lim; i++)
        ans[i] = 0;
    inv(tmp, ans, lim);
    printf("%d\n", ans[n]);
}
void solve11() { printf("%d\n", int(m >= n)); }
void solve12()
{
    printf("%d\n", n - m >= 0 ? ans[n - m] : 0);
```

```
}
int main()
{
    scanf("%d%d", &n, &m);
    fac[0] = 1;
    for (int i = 1; i <= n + m; i++)
        fac[i] = 1ll * fac[i - 1] * i % mod;
    minv[n + m] = pow(fac[n + m], mod - 2);
    for (int i = n + m; i; i--)
        minv[i - 1] = 1ll * minv[i] * i % mod;
    solve1();
    solve2();
    solve3();
    solve4();
    solve5();
    solve6();
    solve7();
    solve8();
    solve9();
    solve10();
    solve11();
    solve12();
    return 0;
}
```

容斥原理

> 定义：$\left|S_1 \cup S_2 \cup S_3 \cup \ldots \cup S_n\right| = \sum_{i=1}^{N} \left|S_i\right| - \sum_{i,j=1}^{N} \left|S_i \cap S_j\right| + \sum_{i,j,k=1}^{N} \left|S_i \cap S_j \cap S_k\right| - \ldots$

例题：给定一个整数 n 和 m 个不同的质数 $p_1, p_2, \ldots, p_m$，请你求出 1 $\sim n$ 中能被 $p_1, p_2, \ldots, p_m$ 中的至少一个数整除的整数有多少个。

二进制枚举解

```
int main(){
    ios::sync_with_stdio(false);cin.tie(0);
    LL n, m;
    cin >> n >> m;
    vector <LL> p(m);
    for (int i = 0; i < m; i ++ )
        cin >> p[i];
    LL ans = 0;
    for (int i = 1; i < (1 << m); i ++ ){
        LL t = 1, cnt = 0;
        for (int j = 0; j < m; j ++ ){
            if (i >> j & 1){
                cnt ++ ;
                t *= p[j];
                if (t > n){
                    t = -1;
                    break;
                }
            }
```

```
        }
        if (t != -1){
            if (cnt & 1) ans += n / t;
            else ans -= n / t;
        }
    }
    cout << ans << "\n";
    return 0;
}
```

dfs 解

```
int main(){
    ios::sync_with_stdio(false);cin.tie(0);
    LL n, m;
    cin >> n >> m;
    vector <LL> p(m);
    for (int i = 0; i < m; i ++ )
        cin >> p[i];
    LL ans = 0;
    function<void(LL, LL, LL)> dfs = [&](LL x, LL s, LL odd){
        if (x == m){
            if (s == 1) return;
            ans += odd * (n / s);
            return;
        }
        dfs(x + 1, s, odd);
        if (s <= n / p[x]) dfs(x + 1, s * p[x], -odd);
    };
    dfs(0, 1, -1);
    cout << ans << "\n";
    return 0;
}
```

康拓展开

正向展开普通解法

将一个字典序排列转换成序号。例如：12345->1，12354->2。

```
int f[20];
void jie_cheng(int n) { // 打出1-n的阶乘表
    f[0] = f[1] = 1; // 0的阶乘为1
    for (int i = 2; i <= n; i++) f[i] = f[i - 1] * i;
}
string str;
int kangtuo() {
    int ans = 1; // 注意，因为 12345 是算作0开始计算的，最后结果要把12345看作是第一个
    int len = str.length();
    for (int i = 0; i < len; i++) {
        int tmp = 0; // 用来计数的
        // 计算str[i]是第几大的数，或者说计算有几个比他小的数
```

```
        for (int j = i + 1; j < len; j++)
            if (str[i] > str[j]) tmp++;
        ans += tmp * f[len - i - 1];
    }
    return ans;
}
int main() {
    jie_cheng(10);
    string str = "52413";
    cout << kangtuo() << endl;
}
```

正向展开树状数组解

给定一个全排列，求出它是 1 ~ n 所有全排列的第几个，答案对 998244353 取模。

答案就是 $\sum_{i=1}^{n} res_{a_i}(n-i)!$ 。res_x 表示剩下的比 x 小的数字的数量，通过**树状数组**处理。

```
#include <bits/stdc++.h>
using namespace std;
#define LL long long
const int mod = 998244353, N = 1e6 + 10;
LL fact[N];
struct fwt{
    LL n;
    vector <LL> a;
    fwt(LL n) : n(n), a(n + 1) {}
    LL sum(LL x){
        LL res = 0;
        for (; x; x -= x & -x)
            res += a[x];
        return res;
    }
    void add(LL x, LL k){
        for (; x <= n; x += x & -x)
            a[x] += k;
    }
    LL query(LL x, LL y){
        return sum(y) - sum(x - 1);
    }
};
int main(){
    ios::sync_with_stdio(false);cin.tie(0);
    LL n;
    cin >> n;
    fwt a(n);
    fact[0] = 1;
    for (int i = 1; i <= n; i ++ ){
        fact[i] = fact[i - 1] * i % mod;
        a.add(i, 1);
    }
    LL ans = 0;
    for (int i = 1; i <= n; i ++ ){
```

```cpp
        LL x;
        cin >> x;
        ans = (ans + a.query(1, x - 1) * fact[n - i] % mod ) % mod;
        a.add(x, -1);
    }
    cout << (ans + 1) % mod << "\n";
    return 0;
}
```

逆向还原

```cpp
string str;
int kangtuo(){
    int ans = 1;  //注意，因为 12345 是算作0开始计算的，最后结果要把12345看作是第一个
    int len = str.length();
    for(int i = 0; i < len; i++){
        int tmp = 0;//用来计数的
        for(int j = i + 1; j < len; j++){
            if(str[i] > str[j]) tmp++;
            //计算str[i]是第几大的数，或者说计算有几个比他小的数
        }
        ans += tmp * f[len - i - 1];
    }
    return ans;
}
int main(){
    jie_cheng(10);
    string str = "52413";
    cout<<kangtuo()<<endl;
}
```

###