

# 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。

```

1  struct Comb {
2      int n;
3      vector<Z> _fac, _inv;
4
5      Comb() : _fac{1}, _inv{0} {}
6      Comb(int n) : Comb() {
7          init(n);
8      }
9      void init(int m) {
10         if (m <= n) return;
11         _fac.resize(m + 1);
12         _inv.resize(m + 1);
13         for (int i = n + 1; i <= m; i++) {
14             _fac[i] = _fac[i - 1] * i;
15         }
16         _inv[m] = _fac[m].inv();
17         for (int i = m; i > n; i--) {
18             _inv[i - 1] = _inv[i] * i;
19         }
20         n = m;
21     }
22     Z fac(int x) {
23         if (x > n) init(x);
24         return _fac[x];
25     }
26     Z inv(int x) {
27         if (x > n) init(x);
28         return _inv[x];
29     }
30     Z C(int x, int y) {
31         if (x < 0 || y < 0 || x < y) return 0;
32         return fac(x) * inv(y) * inv(x - y);
33     }
34     Z P(int x, int y) {
35         if (x < 0 || y < 0 || x < y) return 0;
36         return fac(x) * inv(x - y);
37     }
38 } comb(1 << 21);

```

## 质因数分解

此法适用于： $1 < n, m, MOD < 10^7$  的情况。

```

1  int n,m,p,b[10000005],prime[1000005],t,min_prime[10000005];
2  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
3      for(int i=2;i<=n;i++){
4          if(b[i]==0){
5              prime[++t]=i;
6              min_prime[i]=t;
7          }
8          for(int j=1;j<=t&&i*prime[j]<=n;j++){
9              b[prime[j]*i]=1;
10             min_prime[prime[j]*i]=j;
11             if(i%prime[j]==0) break;
12         }
13     }
14 }
15 long long c(int n,int m,int p){//计算C(n,m)%p的值
16     euler_Prime(n);
17     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
18     for(int i=1;i<=t;i++) a[i]=0;//注意清0，一开始是随机数
19     for(int i=n;i>=n-m+1;i--){//处理分子
20         int x=i;
21         while (x!=1){
22             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）
23             x/=prime[min_prime[x]];
24         }
25     }
26     for(int i=1;i<=m;i++){//处理分母
27         int x=i;
28         while (x!=1){
29             a[min_prime[x]]--;
30             x/=prime[min_prime[x]];
31         }
32     }
33     long long ans=1;
34     for(int i=1;i<=t;i++){//枚举质数的编号，看它出现了几次
35         while(a[i]>0){
36             ans=ans*prime[i]%p;
37             a[i]--;
38         }
39     }
40     return ans;
41 }
42 int main(){
43     cin>>n>>m;
44     m=min(m,n-m);//小优化
45     cout<<c(n,m,MOD);
46 }

```

## 杨辉三角（精确计算）

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

```

1  vector C(n + 1, vector<int>(n + 1));
2  C[0][0] = 1;
3  for (int i = 1; i <= n; i++) {
4      C[i][0] = 1;
5      for (int j = 1; j <= n; j++) {
6          C[i][j] = C[i - 1][j] + C[i - 1][j - 1];
7      }
8  }
9  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

$n \leq m$ ,  $A(m, n)$

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

$n \leq m$ , 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。**

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

```

1  #include <algorithm>
2  #include <cstdio>
3  int n, m, fac[400010], minv[400010];
4  int const mod = 998244353, g = 3, gi = (mod + 1) / g;
5  int C(int x, int y)
6  {
7      if (x < 0 || y < 0 || x < y)
8          return 0;
9      else
10         return 1ll * fac[x] * minv[y] % mod * minv[x - y] % mod;
11 }
12 int pow(int x, int y)
13 {
14     int res = 1;
15     while (y) {
16         if (y & 1)
17             res = 1ll * res * x % mod;
18         x = 1ll * x * x % mod;
19         y >>= 1;
20     }
21     return res;
22 }
23 struct NTT {
24     int r[800010], lim;
25     NTT()
26         : r()
27         , lim()
28     {
29     }
30     void getr(int lm)
31     {

```

```

32     lim = lm;
33     for (int i = 0; i < lim; i++)
34         r[i] = (r[i >> 1] >> 1) | ((i & 1) * (lim >> 1));
35 }
36 void operator()(int* a, int type)
37 {
38     for (int i = 0; i < lim; i++)
39         if (i < r[i])
40             std::swap(a[i], a[r[i]]);
41     for (int mid = 1; mid < lim; mid <= 1) {
42         int rt = pow(type == 1 ? g : gi, (mod - 1) / (mid << 1));
43         for (int j = 0, r = mid << 1; j < lim; j += r) {
44             int p = 1;
45             for (int k = 0; k < mid; k++, p = 1ll * p * rt % mod) {
46                 int x = a[j + k], y = 1ll * a[j + mid + k] * p % mod;
47                 a[j + k] = (x + y) % mod, a[j + mid + k] = (x - y + mod) % mod;
48             }
49         }
50     }
51     if (type == -1)
52         for (int i = 0, p = pow(lim, mod - 2); i < lim; i++)
53             a[i] = 1ll * a[i] * p % mod;
54 }
55 } ntt;
56 void inv(int const* a, int* ans, int n)
57 {
58     static int tmp[800010];
59     for (int i = 0; i < n << 1; i++)
60         tmp[i] = ans[i] = 0;
61     ans[0] = pow(a[0], mod - 2);
62     for (int m = 2; m <= n; m <= 1) {
63         int lim = m << 1;
64         ntt.getr(lim);
65         for (int i = 0; i < m; i++)
66             tmp[i] = a[i];
67         ntt(tmp, 1), ntt(ans, 1);
68         for (int i = 0; i < lim; i++)
69             ans[i] = ans[i] * (2 - 1ll * ans[i] * tmp[i] % mod + mod) % mod, tmp[i] =
0;
70         ntt(ans, -1);
71         for (int i = m; i < lim; i++)
72             ans[i] = 0;
73     }
74 }
75 void inte(int const* a, int* ans, int n)
76 {
77     for (int i = n - 1; i; i--)
78         ans[i] = 1ll * a[i - 1] * pow(i, mod - 2) % mod;
79     ans[0] = 0;
80 }
81 void der(int const* a, int* ans, int n)
82 {
83     for (int i = 1; i < n; i++)

```

```

84     ans[i - 1] = 111 * i * a[i] % mod;
85     ans[n - 1] = 0;
86 }
87 void ln(int const* a, int* ans, int n)
88 {
89     static int b[800010];
90     for (int i = 0; i < n << 1; i++)
91         ans[i] = b[i] = 0;
92     inv(a, ans, n);
93     der(a, b, n);
94     int lim = n << 1;
95     ntt.getr(lim);
96     ntt(b, 1), ntt(ans, 1);
97     for (int i = 0; i < lim; i++)
98         b[i] = 111 * ans[i] * b[i] % mod, ans[i] = 0;
99     ntt(b, -1);
100    for (int i = n; i < lim; i++)
101        b[i] = 0;
102    inte(b, ans, n);
103 }
104 void exp(int const* a, int* ans, int n)
105 {
106     static int f[800010];
107     for (int i = 0; i < n << 1; i++)
108         ans[i] = f[i] = 0;
109     ans[0] = 1;
110     for (int m = 2; m <= n; m <= 1) {
111         int lim = m << 1;
112         ln(ans, f, m);
113         f[0] = (a[0] + 1 - f[0] + mod) % mod;
114         for (int i = 1; i < m; i++)
115             f[i] = (a[i] - f[i] + mod) % mod;
116         ntt.getr(lim);
117         ntt(f, 1), ntt(ans, 1);
118         for (int i = 0; i < lim; i++)
119             ans[i] = 111 * ans[i] * f[i] % mod, f[i] = 0;
120         ntt(ans, -1);
121         for (int i = m; i < lim; i++)
122             ans[i] = 0;
123     }
124 }
125 void solve1() { printf("%d\n", pow(m, n)); }
126 void solve2()
127 {
128     if (m < n)
129         puts("0");
130     else
131         printf("%lld\n", 111 * fac[m] * minv[m - n] % mod);
132 }
133 void solve3()
134 {
135     if (n < m)
136         return puts("0"), void();

```

```

137     int ans = 0;
138     for (int i = 0; i <= m; i++)
139         ans = (ans + 111 * pow(mod - 1, i) * C(m, i) % mod * pow(m - i, n)) % mod;
140     printf("%d\n", ans);
141 }
142 int s[800010];
143 void solve4()
144 {
145     static int tmp[800010];
146     for (int i = 0; i <= n; i++)
147         tmp[i] = (i & 1 ? mod - 111 : 111) * minv[i] % mod, s[i] = 111 * pow(i, n) *
minv[i] % mod;
148     int lim = 1;
149     for (lim = 1; lim <= n + n; lim <= 1)
150         ;
151     ntt.getr(lim);
152     ntt(tmp, 1), ntt(s, 1);
153     for (int i = 0; i < lim; i++)
154         s[i] = 111 * s[i] * tmp[i] % mod;
155     ntt(s, -1);
156     for (int i = n + 1; i < lim; i++)
157         s[i] = 0;
158     int ans = 0;
159     for (int i = 0; i <= m; i++)
160         ans = (ans + s[i]) % mod;
161     printf("%d\n", ans);
162 }
163 void solve5() { printf("%d\n", int(m >= n)); }
164 void solve6() { printf("%d\n", s[m]); }
165 void solve7() { printf("%d\n", C(n + m - 1, m - 1)); }
166 void solve8() { printf("%d\n", C(m, n)); }
167 void solve9() { printf("%d\n", C(n - 1, m - 1)); }
168 int ans[800010];
169 void solve10()
170 {
171     static int tmp[800010];
172     for (int i = 1; i <= m; i++)
173         for (int j = 1; j * i <= n; j++)
174             ans[i * j] = (ans[i * j] - 111 * minv[j] * fac[j - 1] % mod + mod) % mod;
175     int lim = 1;
176     for (; lim <= n; lim <= 1)
177         ;
178
179     exp(ans, tmp, lim);
180     for (int i = 0; i < lim; i++)
181         ans[i] = 0;
182     inv(tmp, ans, lim);
183     printf("%d\n", ans[n]);
184 }
185 void solve11() { printf("%d\n", int(m >= n)); }
186 void solve12()
187 {
188     printf("%d\n", n - m >= 0 ? ans[n - m] : 0);

```

```

189 }
190 int main()
191 {
192     scanf("%d%d", &n, &m);
193     fac[0] = 1;
194     for (int i = 1; i <= n + m; i++)
195         fac[i] = 1ll * fac[i - 1] * i % mod;
196     minv[n + m] = pow(fac[n + m], mod - 2);
197     for (int i = n + m; i; i--)
198         minv[i - 1] = 1ll * minv[i] * i % mod;
199     solve1();
200     solve2();
201     solve3();
202     solve4();
203     solve5();
204     solve6();
205     solve7();
206     solve8();
207     solve9();
208     solve10();
209     solve11();
210     solve12();
211     return 0;
212 }

```

## 容斥原理

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

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

## 二进制枚举解

```

1  int main(){
2      ios::sync_with_stdio(false);cin.tie(0);
3      LL n, m;
4      cin >> n >> m;
5      vector <LL> p(m);
6      for (int i = 0; i < m; i ++ )
7          cin >> p[i];
8      LL ans = 0;
9      for (int i = 1; i < (1 << m); i ++ ){
10         LL t = 1, cnt = 0;
11         for (int j = 0; j < m; j ++ ){
12             if (i >> j & 1){
13                 cnt ++ ;
14                 t *= p[j];
15                 if (t > n){
16                     t = -1;
17                     break;
18                 }
19             }

```



```

20     }
21     if (t != -1){
22         if (cnt & 1) ans += n / t;
23         else ans -= n / t;
24     }
25 }
26 cout << ans << "\n";
27 return 0;
28 }

```

## dfs 解

```

1  int main(){
2      ios::sync_with_stdio(false);cin.tie(0);
3      LL n, m;
4      cin >> n >> m;
5      vector <LL> p(m);
6      for (int i = 0; i < m; i ++ )
7          cin >> p[i];
8      LL ans = 0;
9      function<void(LL, LL, LL)> dfs = [&](LL x, LL s, LL odd){
10         if (x == m){
11             if (s == 1) return;
12             ans += odd * (n / s);
13             return;
14         }
15         dfs(x + 1, s, odd);
16         if (s <= n / p[x]) dfs(x + 1, s * p[x], -odd);
17     };
18     dfs(0, 1, -1);
19     cout << ans << "\n";
20     return 0;
21 }

```

## 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。

```

1  int f[20];
2  void jie_cheng(int n) { // 打出1-n的阶乘表
3      f[0] = f[1] = 1; // 0的阶乘为1
4      for (int i = 2; i <= n; i++) f[i] = f[i - 1] * i;
5  }
6  string str;
7  int kangtuo() {
8      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
9      int len = str.length();
10     for (int i = 0; i < len; i++) {
11         int tmp = 0; // 用来计数的
12         // 计算str[i]是第几大的数，或者说计算有几个比他小的数

```

```

13     for (int j = i + 1; j < len; j++)
14         if (str[i] > str[j]) tmp++;
15     ans += tmp * f[len - i - 1];
16 }
17 return ans;
18 }
19 int main() {
20     jie_cheng(10);
21     string str = "52413";
22     cout << kangtuo() << endl;
23 }

```

## 正向展开树状数组解

给定一个全排列，求出它是  $1 \sim n$  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。

```

1  #include <bits/stdc++.h>
2  using namespace std;
3  #define LL long long
4  const int mod = 998244353, N = 1e6 + 10;
5  LL fact[N];
6  struct fwt{
7      LL n;
8      vector <LL> a;
9      fwt(LL n) : n(n), a(n + 1) {}
10     LL sum(LL x){
11         LL res = 0;
12         for (; x; x -= x & -x)
13             res += a[x];
14         return res;
15     }
16     void add(LL x, LL k){
17         for (; x <= n; x += x & -x)
18             a[x] += k;
19     }
20     LL query(LL x, LL y){
21         return sum(y) - sum(x - 1);
22     }
23 };
24 int main(){
25     ios::sync_with_stdio(false); cin.tie(0);
26     LL n;
27     cin >> n;
28     fwt a(n);
29     fact[0] = 1;
30     for (int i = 1; i <= n; i++){
31         fact[i] = fact[i - 1] * i % mod;
32         a.add(i, 1);
33     }
34     LL ans = 0;
35     for (int i = 1; i <= n; i++){

```

```

36     LL x;
37     cin >> x;
38     ans = (ans + a.query(1, x - 1) * fact[n - i] % mod) % mod;
39     a.add(x, -1);
40 }
41 cout << (ans + 1) % mod << "\n";
42 return 0;
43 }

```

## 逆向还原

```

1  string str;
2  int kangtuo(){
3      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
4      int len = str.length();
5      for(int i = 0; i < len; i++){
6          int tmp = 0; //用来计数的
7          for(int j = i + 1; j < len; j++){
8              if(str[i] > str[j]) tmp++;
9              //计算str[i]是第几大的数，或者说计算有几个比他小的数
10         }
11         ans += tmp * f[len - i - 1];
12     }
13     return ans;
14 }
15 int main(){
16     jie_cheng(10);
17     string str = "52413";
18     cout<<kangtuo()<<endl;
19 }

```

###