

Template For ICPC

--lovekdl

FFT

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  #define rint register int
4  #define ll long long
5  #define rll register long long
6  #define db long double
7  const int N=1<<21;
8  const db pi=acosl(-1);
9  struct cp{
10     db x,y;
11     cp operator + (const cp&A)const{return (cp){x+A.x,y+A.y};}
12     cp operator - (const cp&A)const{return (cp){x-A.x,y-A.y};}
13     cp operator * (const cp&A)const{return (cp){x*A.x-
y*A.y,x*A.y+y*A.x};}
14     cp operator / (const db&A)const{return (cp){x/A,y/A};}
15 }w[N],t;
16 int r[N],pd;
17 void fft(rint n,vector<cp> &a,rint typ){
18     if(pd!=n){
19         for(rint i=0;i<n;i++)
20             r[i]=(r[i>>1]>>1)|((i&1)?n>>1:0);
21         pd=n;
22     }
23     a.resize(n);
24     for(rint i=0;i<n;i++)
25         if(i<r[i])swap(a[i],a[r[i]]);
26     for(rint mid=1;mid<n;mid<=<1)
27         for(rint i=0;i<n;i+=mid<<1)
28             for(rint j=0;j<mid;j++)
29                 t=w[mid+j]*a[i+j+mid],a[i+j+mid]=a[i+j]-
t,a[i+j]=a[i+j]+t;
30     if(~typ)return ;
31     reverse(a.begin()+1,a.end());
32     for(rint i=0;i<n;i++)
33         a[i]=a[i]/n;
34 }
35 void init(){
```

```

36     w[N/2]=(cp){1.0,0.0};w[N/2+1]=t=(cp)
    {cosl(2*pi/N),sinl(2*pi/N)};
37     for(rint i=N/2+2;i<N;i++)w[i]=w[i-1]*t;
38     for(rint i=N/2-1;i;i--)w[i]=w[i<<1];
39 }
40 vector<cp> Mul(vector<cp> a,vector<cp> b){
41     int n=1;
42     while(n<=a.size()+b.size())n<<=1;
43     fft(n,a,1);fft(n,b,1);
44     for(rint i=0;i<n;i++)
45         a[i]=a[i]*b[i];
46     fft(n,a,-1);
47     return a;
48 }
49 inline int read(){
50     int x=0,f=1;char ch=getchar();
51     while(ch<'0' || ch>'9'){if(ch=='-')f=-1;ch=getchar();}
52     while(ch<='9'&&ch>='0'){x=x*10+ch-'0';ch=getchar();}
53     return x*f;
54 }
55 int main(){
56     init();
57     rint n=read(),m=read();
58     ++n;++m;
59     vector<cp> f,g;
60     f.resize(n);g.resize(m);
61     for(rint i=0;i<n;i++)
62         f[i].x=read();
63     for(rint i=0;i<m;i++)
64         g[i].x=read();
65     f=Mul(f,g);
66     for(rint i=0;i<n+m-1;i++)
67         printf("%d ",int(f[i].x+0.3));
68     return 0;
69 }

```

多项式

```
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const int N = 3e5+10;
5  const int mod = 998244353, g = 3, gi = 332748118;
6  //const int mod = 4179340454199820289, g = 3, gi =
   1393113484733273430;
7  int n, m;
8  int a[N], b[N];
9  int re[N];
10
11
12 int ksm(int a, int b) {
13     int ret = 1;
14     while(b) {
15         if(b & 1) ret = ret * a % mod;
16         a = a * a % mod;
17         b >>= 1;
18     }
19     return ret;
20 }
21 int inv(int x) {
22     return ksm(x, mod - 2);
23 }
24
25 void ntt(int *a, int lim, int opt) {
26     for(int i = 0; i < lim; ++i)
27         if(i < re[i]) swap(a[i], a[re[i]]);
28     for(int len = 1; len < lim; len <= 1) {
29         int wn = ksm(opt == 1 ? g : gi, (mod - 1) / (len <<
   1));
30         for(int i = 0; i < lim; i += (len << 1)) {
31             int w = 1;
32             for(int j = 0; j < len; ++j) {
33                 int x = a[i + j], y = w * a[i + j + len] % mod;
34                 a[i + j] = (x + y) % mod;
35                 a[i + j + len] = (x - y + mod) % mod;
```

```

36         w = w * wn % mod;
37     }
38 }
39 }
40 if(opt == 1) return;
41 int limv = inv(lim);
42 for(int i = 0; i < lim; ++i) {
43     a[i] = a[i] * limv % mod;
44 }
45 }
46
47 //n次多项式和m次多项式卷积
48 void mul(int *F, int n, int *G, int m) {
49     // if(n+m < 128) {
50     //     for(int i = 0; i <= n + m; ++i) {
51     //         H[i] = 0;
52     //     }
53     //     for(int i = 0; i <= n; ++i) {
54     //         for(int j = 0; j <= m; ++j) {
55     //             H[i+j] = (H[i+j] + F[i] * G[j] % mod) % mod;
56     //         }
57     //         F[i] = H[i];
58     //     }
59     //     for(int i = n + 1; i <= n + m; ++i) {
60     //         F[i] = H[i];
61     //     }
62     //     return;
63     // }
64     int lim = 1, ti = 0;
65     while(lim <= n + m) {
66         lim <<= 1;
67         ti++;
68     }
69     for(int i = n + 1; i < lim; ++i) F[i] = 0;
70     for(int j = m + 1; j < lim; ++j) G[j] = 0;
71     for(int i = 0; i < lim; ++i) {
72         re[i] = (re[i] >> 1) >> 1 | ((i & 1) << (ti - 1));
73     }
74     ntt(F, lim, 1);
75     ntt(G, lim, 1);
76     for(int i = 0; i < lim; ++i) {
77         F[i] = F[i] * G[i] % mod;

```

```

78     }
79     ntt(F, lim, -1);
80     for(int i = n + m + 1; i < lim; ++i) assert(F[i] == 0);
81 }
82
83 //n-1次多项式求逆
84
85 int H[N];
86 void inv(int *F, int *G, int n) {
87     if(n == 1) {G[0] = inv(F[0]);return;}
88     inv(F, G, (n+1)>>1);
89     int ti = 0, lim = 1;
90     while(lim < n<<1) {
91         lim <= 1;
92         ti++;
93     }
94     for(int i = 1; i < lim; ++i) {
95         re[i] = (re[i] >> 1) >> 1 | ((i & 1) << (ti - 1));
96     }
97     for(int i = 0; i < n; ++i) H[i] = F[i];
98     for(int i = n; i < lim; ++i) H[i] = G[i] = 0;
99     ntt(H, lim, 1);
100    ntt(G, lim, 1);
101    for(int i = 0; i < lim; ++i) {
102        G[i] = G[i]*(211-H[i]*G[i] % mod + mod) % mod;
103    }
104    ntt(G, lim, -1);
105    for(int i = n; i < lim; ++i) G[i] = 0;
106 }
107 //求导, F->G
108 void diff(int *F, int *G, int n) {
109     for(int i = 1; i < n; ++i) G[i-1] = F[i] * i % mod;
110     G[n-1] = 0;
111 }
112 //积分, F->G
113 void integral(int *F, int *G, int n) {
114     for(int i = 1; i < n; ++i) G[i] = F[i - 1] * inv(i) % mod;
115     G[0] = 0;
116 }
117
118 //多项式ln
119 int Fi[N], Fd[N];

```

```

120 void ln(int *F, int *G, int n) {
121     for(int i = 0; i < (n<<2); ++i) G[i] = 0;
122     inv(F, Fi, n);
123     diff(F, Fd, n);
124     mul(Fi, n-1, Fd, n-1);
125     integral(Fi, G, n);
126 }
127 //多项式exp
128 int lnG[N];
129 void exp(int *F, int *G, int n) {
130     if(n == 1) {G[0] = 1;return;}
131     exp(F, G, n + 1 >> 1);
132     assert(G[0] == 1);
133     ln(G, lnG, n);
134     assert(lnG[0] == 0);
135     for(int i = 0; i < n; ++i) lnG[i] = (F[i] - lnG[i] + mod) %
mod;
136
137     lnG[0]++;lnG[0] %= mod;
138     mul(G, n - 1, lnG, n - 1);
139 }
140
141 int G2[N], GG[N], G2i[N];
142 void sqrt(int *F, int *G, int n){
143     if(n == 1){
144         G[0] = 1;
145         return;
146     }
147
148     sqrt(F, G, n + 1 >> 1);
149     for(int i = 0; i < n; ++i) {
150         G2[i] = G[i] * 2 % mod;
151         GG[i] = G[i];
152         G2i[i] = 0;
153     }
154     inv(G2, G2i, n);
155
156     mul(G, n - 1, GG, n - 1);
157     for(int i = 0; i < n; ++i) {
158         G[i] = (G[i] + F[i]) % mod;
159     }
160     mul(G, n - 1, G2i, n - 1);

```

```

161     return;
162 }
163
164 void solve() {
165
166     // int x, y;
167     // cin>>x>>y;
168     // for(int i = 0; i <= x; ++i) {
169     //     cin>>a[i];
170     // }
171     // for(int i = 0; i <= y; ++i) {
172     //     cin>>b[i];
173     // }
174     // mul(a, x, b, y);
175     // for(int i = 0; i <= x + y; ++i) {
176     //     cout<<a[i]<<" ";
177     // }
178
179     // int n;
180     // cin>>n;
181     // for(int i = 0 ; i < n; ++i) {
182     //     cin>>a[i];
183     // }
184     // inv(a, b, n);
185     // for(int i = 0; i < n; ++i) {
186     //     cout<<b[i]<<" ";
187     // }
188
189
190
191     // int n;
192     // cin>>n;
193     // for(int i = 0; i < n; ++i) {
194     //     cin>>a[i];
195     // }
196     // ln(a, b, n);
197     // for(int i =0; i < n; ++i) {
198     //     cout<<b[i]<<" ";
199     // }
200
201     // int n;
202     // cin>>n;

```



```

203     // for(int i = 0; i < n; ++i) {
204     //     cin>>a[i];
205     // }
206     // exp(a, b, n);
207     // for(int i = 0; i < n; ++i) {
208     //     cout<<b[i]<<" ";
209     // }
210
211     int n;
212     cin>>n;
213     for(int i = 0; i < n; ++i) {
214         cin>>a[i];
215     }
216     int sum = 1;
217     while(sum <= n) sum <= 1;
218     sqrt(a, b, sum);
219     for(int i = 0; i < n; ++i) {
220         cout<<b[i]<<" ";
221     }
222 }
223
224
225
226 signed main() {
227     ios::sync_with_stdio(false);
228     cin.tie(nullptr);
229     solve();
230     return 0;
231 }

```

分治NTT

```
1  /*
2  分治ntt
3  计算f[i] = sum(f[i-j] * g[j])
4  */
5
6  #include <bits/stdc++.h>
7  using namespace std;
8  #define int long long
9  const int maxn = 2e6+10;
10 const int mod = 998244353, G = 3, Gi = (mod+1)/3;
11 int r[maxn], a[maxn], b[maxn], f[maxn], g[maxn], n;
12 int qpow(int a, int b) {
13     int ret = 1;
14     while(b) {
15         if(b & 1) ret = ret * a % mod;
16         a = a * a % mod;
17         b >>= 1;
18     }
19     return ret;
20 }
21 void NTT(int *a, int limit, int type) {
22     for(int i = 0; i < limit; i++)
23         if(i < r[i]) swap(a[i], a[r[i]]);
24     for(int mid = 1; mid < limit; mid <= 1) {
25         int wn = qpow((type == 1 ? G : Gi), (mod - 1) / (mid <<
26 1));
27         for(int R = mid << 1, i = 0; i < limit; i += R)
28             for(int k = 0, w = 1; k < mid; k++, w = w * wn % mod) {
29                 int x = a[i+k], y = a[i+k+mid] * w % mod;
30                 a[i+k] = (x + y) % mod, a[i+k+mid] = (x - y + mod) %
31 mod;
32             }
33     }
34     if(type == 1) return;
35     int inv = qpow(limit, mod-2);
36     for(int i = 0; i < limit; i++) a[i] = a[i] * inv % mod;
37 }
38 void mul(int *a, int *b, int limit) {
39     for(int i = 0; i < limit; i++)
```

```

38     r[i] = (r[i >> 1] >> 1) | ((i & 1) ? limit >> 1 : 0);
39     NTT(a, limit, 1); NTT(b, limit, 1);
40     for(int i = 0; i < limit; i++) a[i] = a[i] * b[i] % mod;
41     NTT(a, limit, -1);
42 }
43 void solve(int l, int r) {
44     if(l == r) return;
45     int mid = l + r >> 1;
46     solve(l, mid);
47     int limit = 1;
48     while(limit <= mid - l + r - 1) limit <= 1;
49     for(int i = 0; i < limit; i++) a[i] = b[i] = 0;
50     for(int i = l; i <= mid; i++) a[i - l] = f[i];
51     for(int i = 1; i <= r - l; i++) b[i] = g[i];
52     mul(a, b, limit);
53     for(int i = mid + 1; i <= r; i++) f[i] = (f[i] + a[i -
1])%mod;
54     solve(mid+1, r);
55 }
56 signed main() {
57     cin >> n;
58     for(int i = 1; i < n; i++) scanf("%lld", &g[i]);
59     f[0] = 1;
60     solve(0, n-1);
61     for(int i = 0; i < n; i++) printf("%lld ", f[i]);
62 }

```

$$f(k) = \sum_{i=0}^n y_i \prod_{i \neq j} \frac{k - x[j]}{x[i] - x[j]}$$

对于分子来说，我们维护出关于 k 的前缀积和后缀积，也就是

$$pre_i = \prod_{j=0}^i k - j$$

$$suf_i = \prod_{j=i}^n k - j$$

对于分母来说，观察发现这其实就是阶乘的形式，我们用 $fac[i]$ 来表示 $i!$

那么式子就变成了

$$f(k) = \sum_{i=0}^n y_i \frac{pre_{i-1} * suf_{i+1}}{fac[i-1] * fac[N-i]}$$

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const int N = 1e4+10;
5  const int mod = 998244353;
6  int T;
7  int n, k;
8  int x[N], y[N];
9
10 int qpow(int a, int b) {
11     int ret = 1;
12     while(b) {
13         if(b & 1) ret = ret * a % mod;
14         a = a * a % mod;
15         b >>= 1;
16     }
17     return ret;
18 }
19
20 int lagrange(int k) {
21     int ans = 0;

```

```

22     for(int i = 1; i <= n; ++i) {
23         int now = y[i];
24         for(int j = 1; j <= n; ++j) {
25             if(j == i) continue;
26             now = now * (k - x[j]) % mod * qpow(x[i] - x[j], mod
- 2) %mod;
27         }
28         ans = (ans + now) % mod;
29     }
30     if(ans < 0) ans += mod;
31     return ans;
32 }

```

exgcd

```
1 int exgcd(int a, int b, int &x, int &y) {
2     if (b == 0) {
3         x = 1;
4         y = 0;
5         return a;
6     }
7     int d = exgcd(b, a % b, y, x);
8     y -= (a / b) * x;
9     return d;
10 }
```

合并两个同余方程(CRT)

```
1 //x = a mod b
2 //x = c mod d
3 void merge(ll &a, ll &b, ll c, ll d) {
4     if (a == -1 && b == -1) return;
5     ll x, y;
6     ll g = exgcd(b, d, x, y);
7     if ((c - a) % g != 0) {
8         a = b = -1;
9         return;
10    }
11    d /= g;
12    ll t0 = ((c - a) / g) % d * x % d;
13    if (t0 < 0) t0 += d;
14    a = b * t0 + a;
15    b = b * d;
16 }
```

$$f[n] = \sum_{d|n} g(d) \text{ 求 } g(n)$$

```

1  #include<bits/stdc++.h>
2  #define uint unsigned int
3  using namespace std;
4  typedef long long ll;
5  const int N = 1e6+101;
6  uint f[N];
7  int n;
8  int p[N], pr[N], pe[N];
9  int cnt;
10 uint mu[N];
11 uint g[N];
12 unsigned int A,B,C;
13 void solve() {
14     scanf("%d", &n);
15     for (int i = 1; i <= n; i++)
16         cin>>f[i];
17     p[1] = 1;mu[1] = 1;
18     for(int i = 2; i <= n; ++i) {
19         if(!p[i]) {
20             p[i] = i;
21             pr[++cnt] = i;
22             mu[i] = (uint)-1;
23         }
24         for(int j = 1; j <= cnt && i * pr[j] <= n; ++j) {
25             p[i * pr[j]] = pr[j];
26             if(p[i] == pr[j]) {
27                 mu[i * pr[j]] = 0;
28                 break;
29             }
30             else mu[i * pr[j]] = (uint) -mu[i];
31         }
32     }
33     for(int d1 = 1; d1 <= n; ++d1)
34         for(int d2 = 1; d2 * d1 <= n; ++d2)
35             g[d1 * d2] += f[d1] * mu[d2];
36 }
37 int main() {
38     solve();

```

```
39     return 0;  
40 }
```


求积性函数

```
1 void compute(function<void(int)> calc) {
2     f[1] = 1;
3     uint ans = 0;
4     for(int i = 2; i <= n; ++i) {
5         if(i == pe[i]) calc(i);
6         else f[i] = f[i / pe[i]] * f[pe[i]];
7         ans = ans ^ (a * i * f[i] + b);
8     }
9     ans = ans ^ (a + b);
10    printf("%u\n", ans);
11 }
12
13 void solve() {
14     scanf("%d%u%u", &n, &a, &b);
15     for(int i = 2; i <= n; ++i) {
16         if(!p[i]) {
17             p[i] = i;
18             pe[i] = i;
19             pr[++cnt] = i;
20         }
21         for(int j = 1; j <= cnt && i * pr[j] <= n; ++j) {
22             p[i * pr[j]] = pr[j];
23             if(p[i] == pr[j]) {
24                 pe[i * pr[j]] = pe[i] * pr[j];
25                 break;
26             }
27             else {
28                 pe[i * pr[j]] = pr[j];
29             }
30         }
31     }
```

$O(\sqrt{n})$ 求 $\sum_{i=1}^n \gcd(i, n)$

积性函数, $g(p^a) = (a+1)p^a - ap^{a-1}$

```
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  int g(int n){
5      int ans = 1;
6      for(int i=2;i*i<=n;++i){
7          if(n%i) continue;
8          int a=0;
9          int p=1;
10         while(n%i==0) {
11             p *= i;
12             ++a;
13             n /= i;
14         }
15         ans *= (a+1)*p - p/i*a;
16     }
17     if(n>1){
18         ans *= n+n-1;
19     }
20     return ans;
21 }
22
23 signed main(){
24     int n;
25     cin >> n;
26     cout << g(n) << endl;
27     return 0;
28 }
```

Pollard rho

```
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4
5  typedef __int128 i128;
6  const int N = 1010;
7  int n;
8  int ans[N], tot;
9
10 int randint(int l, int r) {
11     static mt19937 Rand(time(NULL));
12     uniform_int_distribution<int> dis(l, r);
13     return dis(Rand);
14 }
15
16 int ksm(int a, int b, int p) {
17     int ret = 1;
18     while(b) {
19         if(b & 1) ret = (i128) ret * a % p;
20         a = (i128) a * a % p;
21         b >>= 1;
22     }
23     return ret;
24 }
25
26 bool miller_rabin(int n) {
27     if(n < 3 || n % 2 == 0) return n == 2;
28     int a = n - 1, b = 0;
29     while(a % 2 == 0) {
30         a /= 2;
31         b++;
32     }
33     for(int i = 1, j; i <= 10; ++i) {
34         int x = randint(2, n - 1);
35         int v = ksm(x, a, n);
36         if(v == 1) continue;
37
38         for(j = 0; j < b; ++j) { //Ã»ÓÐint
39             if(v == n - 1) break;
```

```

40         v = (i128)v * v % n;
41     }
42     if(j == b) return 0;
43 }
44 return 1;
45 }
46
47 int pollard_rho(int n) {
48     int s = 0, t = 0;
49     int c = randint(1, n - 1);
50     int step = 0, goal = 1;
51     int value = 1;
52     auto f = [&](int x) {
53         return ((i128)x*x+c)%n;
54     };
55     for(goal = 1; ; goal <= 1, s = t, value = 1) {
56         for(step = 1; step <= goal; step++) {
57             t = f(t);
58             value = ((i128)value * abs(t - s)) % n;
59             if(step % 127 == 0) {
60                 int d = __gcd(value, n);
61                 if(d > 1) return d;
62             }
63         }
64         int d = __gcd(value, n);
65         if(d > 1) return d;
66     }
67     return 0;
68 }
69
70 void get_fac(int n) {
71     if(n == 1) return;
72     if(miller_rabin(n)) {
73         ans[++tot] = n;
74         return;
75     }
76     int p = n;
77     while(p == n) p = pollard_rho(n);
78     while((n % p) == 0) n /= p;
79     get_fac(n);
80     get_fac(p);
81 }

```

```

82
83 void solve() {
84     cin>>n;
85     tot = 0;
86     get_fac(n);
87
88     sort(ans + 1, ans + 1 + tot);
89     tot = unique(ans + 1, ans + 1 + tot) - ans - 1;
90     if(ans[tot] == n) cout<<"Prime\n";
91     else cout<<ans[tot]<<"\n";
92 }
93
94 signed main() {
95     ios::sync_with_stdio(false);
96     cin.tie(nullptr);
97
98     int t = 1;
99     cin>>t;
100     while(t-->0) {
101         solve();
102     }
103
104     return 0;
105 }

```

tarjan缩点

```
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const int maxn = 1e5+100;
5  int n, m;
6  int num, ans;
7  int w[maxn];
8  int dfn[maxn], low[maxn], vis[maxn];
9  int belong[maxn], in[maxn], f[maxn];
10 struct Edge {
11     int next[maxn], to[maxn], from[maxn];
12     int head[maxn], cnt;
13     void con(int u, int v) {
14         next[++cnt] = head[u];
15         to[cnt] = v;
16         from[cnt] = u;
17         head[u] = cnt;
18     }
19 }ED1, ED2;
20 stack <int> stu;
21 void tarjan(int x) {
22     low[x] = dfn[x] = ++num;
23     vis[x] = 1;
24     stu.push(x);
25     for(int i = ED1.head[x]; i; i = ED1.next[i]) {
26         int v = ED1.to[i];
27         if(!dfn[v]) {
28             tarjan(v);
29             low[x] = min(low[x], low[v]);
30         }
31         if(vis[v])
32             low[x] = min(low[x], low[v]);
33     }
34     if(dfn[x] == low[x]) {
35         while(stu.top() != x) {
36             int y = stu.top();
37             stu.pop();
38             belong[y] = x;
39             w[x] += w[y];
```

```

40         vis[y] = 0;
41     }
42     belong[x] = x;
43     vis[x] = 0;
44     stu.pop();
45 }
46 }
47 signed main() {
48     cin>>n>>m;
49     for(int i = 1; i <= n; ++i)
50         scanf("%lld", &w[i]);
51     for(int i = 1; i <= m; ++i) {
52         int u, v;
53         scanf("%lld%lld", &u, &v);
54         ED1.con(u, v);
55     }
56     for(int i = 1; i <= n; ++i) {
57         if(!dfn[i]) tarjan(i);
58     }
59     for(int i = 1; i <= m; ++i) {
60         int u = belong[ED1.from[i]], v = belong[ED1.to[i]];
61         if(u == v) continue;
62         ++in[v];
63         ED2.con(u, v);
64     }
65     //topo
66     return 0;
67 }

```

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  #define int long long
4  const int maxn = 2e5 + 10;
5
6  const int B = 60;
7
8
9
10 struct LinearBasis {
11     vector<int> a = vector<int>(B, 0);
12
13
14     bool insert(int x) {
15         for(int i = B - 1; i >= 0; i--) {
16             if(x & (1LL << i)) {
17                 if(a[i] == 0) { a[i] = x; return true; }
18                 x ^= a[i];
19             }
20         }
21         return false;
22     }
23
24
25     int queryMin(int x) {
26         for(int i = B - 1; i >= 0; i--) {
27             x = min(x, x ^ a[i]);
28         }
29         return x;
30     }
31     int queryMax(int x) {
32         for(int i = B - 1; i >= 0; i--) {
33             x = max(x, x ^ a[i]);
34         }
35         return x;
36     }
37 };
38
39 void work() {
```



```

40     int n, k;
41     cin >> n >> k;
42
43     vector<int> a(n + 1);
44     for(int i = 1; i <= n; i++) cin >> a[i];
45
46     LinearBasis b;
47     int cnt0 = 0;
48     for(int i = 1; i <= n; i++) {
49         if(!b.insert(a[i])) {
50             cnt0++;
51         }
52     }
53
54     // 为什么是向下取整呢？因为有第0小的数，所以是向下取整
55     k = k / (1LL << cnt0);
56     // k >= cnt0;
57
58     int ans = 0;
59     int cnt = 0;
60     for(int i = 0; i < B; i++) {
61         if(b.a[i] == 0) continue;
62         cnt++;
63     }
64     cnt--;
65     for(int i = B - 1; i >= 0; i--) {
66         if(b.a[i] == 0) continue;
67         if(k >= (1LL << cnt)) {
68             k -= 1LL << cnt;
69             ans = max(ans, ans ^ b.a[i]);
70         } else {
71             ans = min(ans, ans ^ b.a[i]);
72         }
73         cnt--;
74     }
75
76     cout << ans << endl;
77
78 }
79
80 signed main() {
81     ios::sync_with_stdio(0);

```

```
82     cin.tie(0);
83
84     int t = 1;
85     // cin >> t;
86     while(t--) {
87         work();
88     }
89
90     return 0;
91 }
```

```
1  #include<bits/stdc++.h>
2  #define int long long
3  #define PII pair<int, int>
4  using namespace std;
5  const int N = 3e5 + 1010;
6  const int inf = 1ll<< 60;
7  int n, m, k;
8  int x[N], tot;
9  int w[N], va[N], dep[N], minp[N], dfn[N], cnt;
10 int lc[N][19];
11 vector<PII> e[N];
12 vector<int> e1[N];
13
14 void dfs1(int x, int fa) {
15     dfn[x] = ++tot;
16     dep[x] = dep[fa] + 1;
17     lc[x][0] = fa;
18     for(auto t : e[x]) {
19         int v = t.first;
20         if(v == fa) continue;
21         minp[v] = min(minp[x], t.second);
22         dfs1(v, x);
23     }
24 }
25 bool cmp(int a, int b) {
26     return dfn[a] < dfn[b];
27 }
28 int lca(int x, int y) {
29     int flag = 0;
30     if(x == 8 && y == 3) {
31         flag = 1;
32     }
33     if(dep[x] < dep[y]) swap(x, y);
34     for(int j = 18; j >= 0; --j) {
35         if(dep[lc[x][j]] >= dep[y]) {
36             x = lc[x][j];
37         }
38     }
39 }
```

```

40     if(x == y) return x;
41     for(int j = 18; j >= 0; --j) {
42         if(lc[x][j] == lc[y][j]) continue;
43         x = lc[x][j];
44         y = lc[y][j];
45     }
46     return lc[x][0];
47 }
48 void con(int x, int y) {
49     e1[x].push_back(y);
50     e1[y].push_back(x);
51 }
52 int stac[N], top = 0;
53 //建树
54 void build() {
55     sort(w + 1, w + 1 + k, cmp);
56     top = 0;
57     stac[++top] = w[1];
58     for(int i = 2; i <= k; ++i) {
59         int lc = lca(stac[top], w[i]);
60         while(dep[stac[top - 1]] >= dep[lc]) {
61             con(stac[top], stac[top - 1]);
62             top--;
63         }
64         if(stac[top] != lc) {
65             con(lc, stac[top]);
66             stac[top] = lc;
67         }
68         stac[++top] = w[i];
69     }
70     for(int i = top; i >= 2; --i) {
71         con(stac[i], stac[i - 1]);
72     }
73 }
74 //dp
75 int dfs(int x, int fa) {
76     int now = 0;
77     for(auto v : e1[x]) {
78         if(v == fa) continue;
79         now += dfs(v, x);
80     }
81     //记得清除虚树

```

```

82     e1[x].clear();
83     if(va[x]) {
84         va[x] = 0;
85         return minp[x];
86     }
87     else return min(now, minp[x]);
88 }
89 void solve() {
90     cin>>n;
91     for(int i = 1, u, v, d; i < n; ++i) {
92         cin>>u>>v>>d;
93         e[u].push_back({v, d});
94         e[v].push_back({u, d});
95     }
96
97     for(int i = 1; i <= n; ++i) minp[i] = inf;
98     minp[1] = inf;
99     dfs1(1, 0);
100    for(int j = 1; j <= 18; ++j) {
101        for(int i = 1; i <= n; ++i) {
102            lc[i][j] = lc[lc[i][j-1]][j-1];
103        }
104    }
105    cin>>m;
106    for(int i = 1; i <= m; ++i) {
107        cin>>k;
108        for(int i = 1; i <= k; ++i) {
109            cin>>w[i];
110            va[w[i]] = 1;
111        }
112        build();
113        cout<<dfs(stac[1], 0)<<endl;
114    }
115 }
116
117 signed main() {
118     ios::sync_with_stdio(false);
119     cin.tie(nullptr);
120
121     solve();
122     return 0;
123 }

```


输入格式

第一行三个整数 n, m, k 。

接下来 m 行，每行四个整数 x, y, l, r ，表示有一条连接 x, y 的边在 l 时刻出现 r 时刻消失。

输出格式

k 行，第 i 行一个字符串 `Yes` 或 `No`，表示在第 i 时间段内这个图是否是二分图。

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4
5  const int N = 5e5 + 1010;
6  const int inf = 1ll << 60;
7  int n, m, k;
8  int a[N];
9  int stac[N];
10
11 vector<int> in[N];
12 vector<int> out[N];
13 int mark[N];
14 struct ED {
15     int u, v, l, r;
16 }edge[N];
17
18 struct node{
19     int son[2], fa, val, sum;
20     int st;
21     int mint, minid;
22     int flag;
23 }t[N];
24
25 void update(int x) {
26     //t[x].sum = t[t[x].son[0]].sum ^ t[t[x].son[1]].sum ^
    t[x].val;
27     t[x].sum = t[t[x].son[0]].sum + t[t[x].son[1]].sum +
    t[x].val;
28     t[x].mint = t[x].st;

```

```

29     t[x].minid = x;
30     if(t[x].son[0] && t[t[x].son[0]].mint < t[x].mint) {
31         t[x].mint = t[t[x].son[0]].mint;
32         t[x].minid = t[t[x].son[0]].minid;
33     }
34     if(t[x].son[1] && t[t[x].son[1]].mint < t[x].mint) {
35         t[x].mint = t[t[x].son[1]].mint;
36         t[x].minid = t[t[x].son[1]].minid;
37     }
38 }
39
40 void lazy(int x) {
41     swap(t[x].son[0], t[x].son[1]);
42     t[x].flag ^= 1;
43 }
44
45 void pushdown(int x) {
46     if(!t[x].flag) return;
47     lazy(t[x].son[0]);
48     lazy(t[x].son[1]);
49     t[x].flag = 0;
50 }
51
52 bool isroot(int x) {
53     return (t[t[x].fa].son[0] != x && t[t[x].fa].son[1] != x);
54 }
55
56 void rotate(int x){
57     int y = t[x].fa, z = t[y].fa;
58     int tag = (t[y].son[1] == x);
59     if(!isroot(y)) t[z].son[t[z].son[1]==y] = x;
60     t[x].fa = z;
61     t[y].son[tag] = t[x].son[tag^1];
62     t[t[x].son[tag^1]].fa = y;
63     t[x].son[tag^1] = y;
64     t[y].fa = x;
65     update(y);update(x);
66 }
67
68 void splay(int x) {
69     int ptr = 0, y = x;
70     stac[ptr++] = y;

```



```

71     while(!isroot(y)) {
72         stac[ptr++] = t[y].fa;
73         y = t[y].fa;
74     }
75     while(ptr--) pushdown(stac[ptr]);
76     while(!isroot(x)) {
77         int y = t[x].fa, z = t[y].fa;
78         if(!isroot(y)) {
79             (t[y].son[0] == x) ^ (t[z].son[0] == y) ? rotate(x)
: rotate(y);
80         }
81         rotate(x);
82     }
83     update(x);
84 }
85
86
87 void access(int x) {
88     int tp = x, y = 0;
89     while(x) {
90         splay(x);
91         t[x].son[1] = y;
92         update(x);
93         y = x;
94         x = t[x].fa;
95     }
96     splay(tp);
97 }
98
99 void makeroot(int x) {
100     access(x);
101     lazy(x);
102 }
103
104 int findroot(int x) {
105     access(x);
106     while(t[x].son[0]) {
107         pushdown(x);
108         x = t[x].son[0];
109     }
110     splay(x);
111     return x;

```

```

112 }
113
114 void split(int x, int y) {
115     makeroot(x);
116     access(y);
117 }
118
119 void link(int x, int y) {
120     makeroot(x);
121     if(findroot(y) != x) {
122         t[x].fa = y;
123         access(y);
124     }
125 }
126 void cut(int x, int y) {
127     if(findroot(x) != findroot(y)) return;
128     split(x, y);
129     if(t[y].son[0] == x && t[x].son[1] == 0) {
130         t[y].son[0] = 0;
131         t[x].fa = 0;
132         update(y);
133     }
134 }
135 void solve() {
136     cin>>n>>m>>k;
137
138     for(int i = 1; i <= n; ++i) {
139         t[i].st = t[i].mint = inf;
140         t[i].minid = i;
141         t[i].val = t[i].sum = 0;
142     }
143     for(int i = 1, x, y, l, r; i <= m; ++i) {
144         cin>>x>>y>>l>>r;
145         edge[i].u = x;
146         edge[i].v = y;
147         edge[i].l = l;
148         edge[i].r = r;
149         in[l].push_back(i);
150         out[r].push_back(i);
151         t[i + n].mint = r;
152         t[i + n].st = r;
153         t[i + n].minid = i + n;

```

```

154         t[i + n].val = t[i + n].sum = 1;
155     }
156     int ans = 0;
157     for(int i = 0; i < k; ++i) {
158
159         for(auto j : in[i]) {
160             int u = edge[j].u, v = edge[j].v;
161             if(findroot(u) == findroot(v)) {
162                 split(u, v);
163                 int mid = t[v].minid, mint = t[v].mint;
164                 int sum = t[v].sum;
165                 if(edge[j].r <= mint) {
166                     if(sum % 2 == 0) {ans++;mark[j] = 1;}
167                     continue;
168                     //mark
169                 }
170                 if(sum%2 == 0) {
171                     mark[mid - n] = 1;
172                     ans++;
173                 }
174                 cut(edge[mid - n].u, mid);
175                 cut(edge[mid - n].v, mid);
176             }
177             link(u, j + n);
178             link(v, j + n);
179         }
180         for(auto j : out[i]) {
181             int u = edge[j].u, v = edge[j].v;
182             cut(u, j + n);
183             cut(v, j + n);
184             if(mark[j]) ans--;
185         }
186         if(ans) {
187             cout<<"No\n";
188         }
189         else cout<<"Yes\n";
190     }
191     return;
192 }
193
194 signed main() {
195     ios::sync_with_stdio(false);

```

```
196     cin.tie(nullptr);
197
198     int t = 1;
199     //cin>>t;
200     while(t--) {
201         solve();
202     }
203
204     return 0;
205 }
```

现在，你的任务就是随着边的添加，动态的回答小强对于某些边的负载的询问。

输入格式

第一行包含两个整数 N, Q , 表示星球的数量和操作的数量。星球从 1 开始编号。

接下来的 Q 行，每行是如下两种格式之一：

- `A x y` 表示在 x 和 y 之间连一条边。保证之前 x 和 y 是不联通的。
- `Q x y` 表示询问 (x, y) 这条边上的负载。保证 x 和 y 之间有一条边。

```

1  #include<bits/stdc++.h>
2  #define lson t[x].son[0]
3  #define rson t[x].son[1]
4  #define int long long
5  using namespace std;
6
7  const int N = 2e5 + 1010;
8  const int inf = 1ll << 60;
9  int n, q;
10 int a[N];
11 int stac[N];
12
13 int mark[N];
14 struct ED {
15     int u, v, l, r;
16 }edge[N];
17
18 struct node{
19     int son[2], fa;
20     int st;
21     int sz, sz2;
22     int flag;
23 }t[N];
24
25 void update(int x) {
26     t[x].sz = t[lson].sz + t[rson].sz + t[x].sz2 + 1;
27 }
28

```

```

29 void lazy(int x) {
30     swap(t[x].son[0], t[x].son[1]);
31     t[x].flag ^= 1;
32 }
33
34 void pushdown(int x) {
35     if(!t[x].flag) return;
36     lazy(t[x].son[0]);
37     lazy(t[x].son[1]);
38     t[x].flag = 0;
39 }
40
41 bool isroot(int x) {
42     return (t[t[x].fa].son[0] != x && t[t[x].fa].son[1] != x);
43 }
44
45 void rotate(int x){
46     int y = t[x].fa, z = t[y].fa;
47     int tag = (t[y].son[1] == x);
48     if(!isroot(y)) t[z].son[t[z].son[1]==y] = x;
49     t[x].fa = z;
50     t[y].son[tag] = t[x].son[tag^1];
51     t[t[x].son[tag^1]].fa = y;
52     t[x].son[tag^1] = y;
53     t[y].fa = x;
54     update(y);update(x);
55 }
56
57 void splay(int x) {
58     int ptr = 0, y = x;
59     stac[ptr++] = y;
60     while(!isroot(y)) {
61         stac[ptr++] = t[y].fa;
62         y = t[y].fa;
63     }
64     while(ptr--) pushdown(stac[ptr]);
65     while(!isroot(x)) {
66         int y = t[x].fa, z = t[y].fa;
67         if(!isroot(y)) {
68             (t[y].son[0] == x) ^ (t[z].son[0] == y) ? rotate(x)
: rotate(y);
69         }

```

```

70         rotate(x);
71     }
72     update(x);
73 }
74
75
76 void access(int x) {
77     int tp = x, y = 0;
78     while(x) {
79         splay(x);
80         t[x].sz2 += t[rson].sz - t[y].sz;
81         t[x].son[1] = y;
82         update(x);
83         y = x;
84         x = t[x].fa;
85     }
86     splay(tp);
87 }
88
89 void makeroot(int x) {
90     access(x);
91     lazy(x);
92 }
93
94 int findroot(int x) {
95     access(x);
96     while(t[x].son[0]) {
97         pushdown(x);
98         x = t[x].son[0];
99     }
100     splay(x);
101     return x;
102 }
103
104 void split(int x, int y) {
105     makeroot(x);
106     access(y);
107 }
108
109 void link(int x, int y) {
110     makeroot(x);
111     if(findroot(y) != x) {

```

```

112         t[x].fa = y;
113         t[y].sz2 += t[x].sz;
114         //update(y);
115         access(y);
116     }
117
118 }
119
120 void cut(int x, int y) {
121     if(findroot(x) != findroot(y)) return;
122     split(x, y);
123     if(t[y].son[0] == x && t[x].son[1] == 0) {
124         t[y].son[0] = 0;
125         t[x].fa = 0;
126         update(y);
127     }
128 }
129
130 void solve() {
131     cin>>n>>q;
132     for(int i = 1; i <= n; ++i) {
133         t[i].sz = 1;
134     }
135     for(int i = 1; i <= q; ++i) {
136         char op;
137         int x, y;
138         cin>>op>>x>>y;
139         if(op == 'A') {
140             link(x, y);
141         }
142         else {
143             split(x, y);
144             cout<<(t[x].sz2+1) * (t[y].sz2 + 1)<<"\n";
145         }
146     }
147     return;
148 }
149
150
151 signed main() {
152     ios::sync_with_stdio(false);
153     cin.tie(nullptr);

```



```
154
155     int t = 1;
156     //cin>>t;
157     while(t-->0) {
158         solve();
159     }
160
161     return 0;
162 }
```

树上背包

```
1  #include<bits/stdc++.h>
2  using namespace std;
3  typedef long long ll;
4  const int N = 3030;
5  const ll inf = 1<<29;
6  int n, q;
7  int fa[N], a[N], sz[N];
8  ll f[N][N], tmp[N];
9  vector<int> e[N];
10
11 void dfs(int x) {
12     sz[x] = 1;
13     for(auto v : e[x])
14         dfs(v);
15     f[x][1] = a[x];
16     for(auto v : e[x]) {
17         for(int i = 1; i <= sz[x] + sz[v]; ++i) tmp[i] = -inf;
18         for(int i = 1; i <= sz[x]; ++i)
19             for(int j = 0; j <= sz[v]; ++j)
20                 tmp[i + j] = max(tmp[i + j], f[x][i] + f[v][j]);
21         for(int i = 1; i <= sz[x] + sz[v]; ++i)
22             f[x][i] = tmp[i];
23         sz[x] += sz[v];
24     }
25 }
26 int main() {
27     scanf("%d%d", &n, &q);
28     for (int i = 2; i <= n; ++i) {
29         scanf("%d", &fa[i]);
30         e[fa[i]].push_back(i);
31     }
32     for(int i = 1; i <= n; ++i) scanf("%d", &a[i]);
33     dfs(1);
34     for(int i = 1, u, m; i <= q; ++i) {
35         scanf("%d%d", &u, &m);
36         printf("%lld\n", f[u][m]);
37     }
38     return 0;
39 }
```


$n=100$ $m=500$ $k=10$

给定一个包含 n 个结点和 m 条带权边的无向连通图 $G = (V, E)$ 。

再给定包含 k 个结点的点集 S ，选出 G 的子图 $G' = (V', E')$ ，使得：

1. $S \subseteq V'$;
2. G' 为连通图;
3. E' 中所有边的权值和最小。

你只需要求出 E' 中所有边的权值和。

对于 i 的度数为 1 的情况，可以考虑枚举树上与 i 相邻的点 j ，则：

$$dp(j, S) + w(j, i) \rightarrow dp(i, S)$$

对于 i 的度数大于 1 的情况，可以划分成几个子树考虑，即：

$$dp(i, T) + dp(i, S - T) \rightarrow dp(i, S) \quad (T \subseteq S)$$

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const int N = 105;
5  const int inf = 1ll << 60;
6  int n, m, k;
7  int po[N];
8  vector<array<int, 2>> e[N];
9  int dp[N][2020];
10 int vis[N];
11
12 priority_queue<pair<int, int>> q;
13
14 void dij(int state) {
15     for(int i = 1; i <= n; ++i) vis[i] = 0;
16     while(!q.empty()) {
17         int x = q.top().second;
18         q.pop();

```

```

19         if(vis[x]) continue;
20         vis[x] = 1;
21         for(auto item : e[x]) {
22             int v = item[0], w = item[1];
23             if(dp[v][state] > dp[x][state] + w) {
24                 dp[v][state] = dp[x][state] + w;
25                 q.push({-dp[v][state], v});
26             }
27         }
28     }
29 }
30
31 void solve() {
32     cin>>n>>m>>k;
33     for(int i = 1, u, v, w; i <= m; ++i) {
34         cin>>u>>v>>w;
35         e[u].push_back({v, w});
36         e[v].push_back({u, w});
37     }
38     for(int i = 1; i <= n; ++i) {
39         for(int j = 0; j <= ((1ll << k) - 1); ++j) {
40             dp[i][j] = inf;
41         }
42     }
43     for(int i = 1; i <= k; ++i) {
44         cin>>po[i];
45         dp[po[i]][1ll << i-1] = 0;
46     }
47     for(int j = 0; j <= ((1ll << k) - 1); ++j) {
48         for(int i = 1; i <= n; ++i) {
49             for(int k = ((j - 1) & j); k; k = (k - 1) & j) {
50                 dp[i][j] = min(dp[i][j], dp[i][k] + dp[i][j ^
k]);
51             }
52             if(dp[i][j] != inf) q.push({-dp[i][j], i});
53         }
54         dij(j);
55     }
56     int ans = inf;
57     for(int i = 1; i <= n; ++i) ans = min(ans, dp[i][((1ll <<
k) - 1)]);
58     cout<<ans;

```

```
59
60
61 }
62
63 signed main() {
64     ios::sync_with_stdio(false);
65     cin.tie(nullptr);
66     int T = 1;
67     //cin>>T;
68     while(T--) {
69         solve();
70     }
71
72
73     return 0;
74 }
```

区间合并、区间求交

```
1 void merge(vector<PII> &segs) {
2     if (segs.empty()) return;
3     vector<PII> res;
4     sort(segs.begin(), segs.end());
5     int st = segs[0].l, ed = segs[0].r;
6     for (auto seg : segs) {
7         if (seg.l > ed) {
8             res.push_back({st, ed});
9             st = seg.l, ed = seg.r;
10        }
11        else ed = max(ed, seg.r);
12    }
13    res.push_back({st, ed});
14    segs = res;
15 }
16
17 vector<PII> intersection(vector<PII> a, vector<PII> b) {
18     vector<PII> res;
19     int i = 0, j = 0;
20     while (i < a.size() && j < b.size()) {
21         int l = max(a[i].l, b[j].l);
22         int r = min(a[i].r, b[j].r);
23         if (l <= r) res.push_back({l, r});
24         if (a[i].r < b[j].r) i++;
25         else j++;
26     }
27     return res;
28 }
```

子集反演

$$g(S) = \sum_{T \subseteq S} f(T)$$

由子集反演可得：

$$f(S) = \sum_{T \subseteq S} (-1)^{|S|-|T|} g(T)$$

■

```
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  const int N = 520;
5  const int mod = 1e9 + 7;
6  int n;
7  int a[N][N];
8  int sum[1 << 20];
9  vector<pair<int, int>> road[N];
10 int po[N], cnt;
11
12 void clear() {
13     for(int i = 1; i <= n + 1; ++i)
14         for(int j = 1; j <= n + 1; ++j) a[i][j] = 0;
15 }
16
17 void con(int u, int v) {
18     a[u][u]++;
19     a[v][v]++;
20     a[u][v]--;
21     a[v][u]--;
22 }
23
24 void add() {
25     for(int i = 1; i <= cnt; ++i) {
26         for(auto x : road[po[i]]) {
27             con(x.first, x.second);
28         }
29     }
30 }
31
```



```

32 int work() {
33     int w = n;
34     int ret = 1; /*
35     for(int i = 1; i <= w; ++i) {
36         for(int j = 1; j <= w; ++j) cout<<a[i][j]<<" ";
37         cout<<endl;
38     }
39     cout<<endl; */
40     for(int i = 1; i <= w; ++i) {
41         for(int j = i + 1; j <= w; ++j) {
42             while(a[j][i]) {
43                 int l = a[i][i] / a[j][i];
44                 for(int k = i; k <= w; ++k)
45                     a[i][k] = (a[i][k] - l * a[j][k] % mod +
mod) % mod;
46                 for(int k = i; k <= w; ++k)
47                     swap(a[i][k], a[j][k]);
48                 ret = ret * -1;
49             }
50         }
51     }
52     for(int i = 1; i <= w; ++i) {
53         ret = (ret * a[i][i] % mod + mod) % mod;
54     }
55
56     return ret;
57 }
58
59 void solve() {
60     cin>>n;
61     for(int i = 1, x; i < n; ++i) {
62         cin>>x;
63         for(int j = 1, u, v; j <= x; ++j) {
64             cin>>u>>v;
65             road[i].push_back({u, v});
66         }
67     }
68
69     n--;
70     int maxn = (1 << n) - 1;
71     int ans = 0;
72     for(int i = 0; i <= maxn; ++i) {

```

```

73     sum[i] = sum[i>>1] + (i & 1);
74     cnt = 0;
75     for(int j = 0; j <= 20; ++j) {
76         if(i & (1 << j)) po[++cnt] = j + 1;
77     }
78     clear();
79     add();
80     int f = work();
81     //cout<<((n - sum[i]) % 2 ? -1 : 1) * f<<endl;
82     ans = (ans + ((n - sum[i]) % 2 ? -1 : 1) * f % mod +
mod) % mod;
83     }
84     cout<<ans;
85 }
86
87 signed main() {
88     ios::sync_with_stdio(false);
89     cin.tie(NULL);
90     solve();
91     return 0;
92 }

```

子集dp

```

1  for(int i=0;i<w;++i)//依次枚举每个维度{
2      for(int j=0;j<(1<<w);++j)//求每个维度的前缀和{
3          if(j&(1<<i))s[j]+=s[j^(1<<i)];
4      }
5  }

```

And卷积

给两个长度为 2^n 的数组 $f_0, f_1, \dots, f_{2^n-1}, g_0, g_1, \dots, g_{2^n-1}$ 。

求 $h_0, h_1, \dots, h_{2^n-1}$, 满足

$$h_i = \left(\sum_{j \& k = i} f_j \cdot g_k \right) \bmod 10^9 + 7$$

```
1 #include<bits/stdc++.h>
```

```

2  using namespace std;
3  const int N = 2e7+1010;
4  const int inf = 1ll << 60;
5  const int mod = 1e9 + 7;
6  int n, m, k, cnt, len, p, q;
7  int a[N], b[N];
8  string s;
9  long long f[N], g[N], F[N], G[N], H[N];
10 unsigned int A,B,C;
11 inline unsigned int rng61() {
12     A ^= A << 16;
13     A ^= A >> 5;
14     A ^= A << 1;
15     unsigned int t = A;
16     A = B;
17     B = C;
18     C ^= t ^ A;
19     return C;
20 }
21
22 void solve() {
23
24     cin>>n>>A>>B>>C;
25     for (int i = 0; i < (1 << n); i++)
26         F[i] = f[i] = rng61() %mod;
27     for (int i = 0; i < (1 << n); i++)
28         G[i] = g[i] = rng61() % mod;
29     for(int i = 0; i < n; ++i) {
30         for(int j = 0; j < (1ll << n); ++j) {
31             if((j & (1ll << i)) == 0) {
32                 F[j] = F[j] + F[j + (1ll << i)];
33                 G[j] = G[j] + G[j + (1ll << i)];
34             }
35         }
36     }
37     for(int j = 0; j < (1ll << n); ++j) {
38         F[j] %= mod;
39         G[j] %= mod;
40         H[j] = F[j] * G[j] % mod;
41         //if(H[j]) cout<<j<<" "<<F[j]<<endl;
42     }
43     for(int i = 0; i < n; ++i) {

```

```

44         //for(int j = (1ll << n) - 1; j >= 0; --j) {
45         for(int j = 0; j < (1ll << n); ++j) {
46             if((j & (1ll << i)) == 0) {
47                 H[j] -= H[j + (1ll << i)];
48                 H[j] %= mod;
49                 if(H[j] < 0) H[j] += mod;
50             }
51         }
52     }
53     long long ans = 0;
54     for(int j = 0; j < (1ll << n); ++j) {
55         ans ^= H[j];
56     }
57     cout<<ans;
58 }
59
60 signed main() {
61     ios::sync_with_stdio(false);
62     cin.tie(nullptr);
63     int T = 1;
64     //cin>>T;
65     while(T--) {
66         solve();
67     }
68
69
70     return 0;
71 }

```

矩形面积并

```
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4
5  const int N = 2e5 + 1010;
6  const int inf = 1ll << 60;
7
8  int n;
9
10 vector<array<int, 4>> event;
11 vector<int> vx;
12
13 struct node {
14     int mincnt, minv;
15     int flag, tag;
16 }t[N << 4];
17
18 void update(int p) {
19     if(t[p << 1].minv == t[p << 1 | 1].minv) {
20         t[p].mincnt = t[p << 1].mincnt + t[p << 1 | 1].mincnt;
21         t[p].minv = t[p << 1].minv;
22     }
23     else if(t[p << 1].minv < t[p << 1 | 1].minv) {
24         t[p].minv = t[p << 1].minv;
25         t[p].mincnt = t[p << 1].mincnt;
26     }
27     else {
28         t[p].minv = t[p << 1 | 1].minv;
29         t[p].mincnt = t[p << 1 | 1].mincnt;
30     }
31 }
32 void settag(int p, int val){
33     t[p].tag += val;
34     t[p].minv += val;
35 }
36 void pushdown(int p) {
37     int val = t[p].tag;
38     settag(p << 1, val);
39     settag(p << 1 | 1, val);
```

```

40     t[p].tag = 0;
41 }
42
43
44
45 void build(int p, int l, int r) {
46     if(l == r) {
47         t[p].minv = t[p].flag = 0;
48         t[p].mincnt = vx[r] - vx[r - 1];
49         return;
50     }
51     int mid = l + r >> 1;
52     build(p << 1, l, mid);
53     build(p << 1 | 1, mid + 1, r);
54     update(p);
55 }
56
57 void insert(int p, int l, int r, int ql, int qr, int val){
58     if(ql <= l && r <= qr) {
59         settag(p, val);
60         return;
61     }
62     pushdown(p);
63     int mid = l + r >> 1;
64     if(ql <= mid) insert(p << 1, l, mid, ql, qr, val);
65     if(qr > mid) insert(p << 1 | 1, mid + 1, r, ql, qr, val);
66     update(p);
67 }
68
69 void solve() {
70     cin>>n;
71     for(int i = 1, x1, x2, y11, y2; i <= n; ++i) {
72         cin>>x1>>x2>>y11>>y2;
73         vx.push_back(x1);
74         vx.push_back(x2);
75         event.push_back({y11, 1, x1, x2});
76         event.push_back({y2, -1, x1, x2});
77     }
78     sort(event.begin(), event.end());
79     sort(vx.begin(), vx.end());
80     vx.erase(unique(vx.begin(), vx.end()), vx.end());
81     int m = vx.size() - 1;

```

```

82     build(1, 1, m);
83     int ans = 0, prey = 0;
84     int totlen = t[1].mincnt; //0
85     for(auto evt : event) {
86         int cov = totlen;
87         if(t[1].minv == 0) {
88             cov -= t[1].mincnt;
89         }
90         ans += cov * (evt[0] - prey);
91         prey = evt[0];
92         int x1 = lower_bound(vx.begin(), vx.end(), evt[2]) -
vx.begin() + 1;
93         int x2 = lower_bound(vx.begin(), vx.end(), evt[3]) -
vx.begin();
94         if(x1 > x2) continue;
95         insert(1, 1, m, x1, x2, evt[1]);
96     }
97     cout<<ans<<"\n";
98 }
99
100 signed main() {
101     ios::sync_with_stdio(false);
102     cin.tie(nullptr);
103
104     int t = 1;
105     //cin>>t;
106     while(t--) {
107         solve();
108     }
109
110     return 0;
111 }

```

```
1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4
5  const int N = 1e5 + 1010;
6  const int inf = 1ll << 60;
7  const int sc = 5e4;
8  const double eps = 1e-5;
9  int n;
10
11 struct node{
12     double k, b;
13     int flag = 0;
14 }t[N<<3];
15
16 double calc(node line, int x) {
17     return line.k * x + line.b;
18 }
19
20 double cross(node l1, node l2) {
21     return (l1.b - l2.b) / (l2.k - l1.k);
22 }
23
24 void insert(int p, int l, int r, int ql, int qr, node line) {
25     if(ql <= l && r <= qr) {
26         if(!t[p].flag) {
27             t[p] = line;
28             return;
29         }
30         double del1 = calc(line, l) - calc(t[p], l);
31         double del2 = calc(line, r) - calc(t[p], r);
32         if(del1 > eps && del2 > eps) {
33             t[p] = line;
34             return;
35         }
36         if(del1 < eps && del2 < eps) return;
37
38         int mid = l + r >> 1;
39         if(calc(line, mid) - calc(t[p], mid) > eps) {
```



```

40         swap(t[p], line);
41     }
42     double cr = cross(t[p], line);
43     if ((double) mid - cr > eps) {
44         insert(p << 1, l, mid, ql, qr, line);
45     }
46     else {
47         insert(p << 1 | 1, mid + 1, r, ql, qr, line);
48     }
49     return;
50 }
51 int mid = l + r >> 1;
52 if(ql <= mid) insert(p << 1, l, mid, ql, qr, line);
53 if(qr > mid) insert(p << 1 | 1, mid + 1, r, ql, qr, line);
54 return;
55
56 }
57
58 double query(int p, int l, int r, int qx) {
59     if(l == r) {
60         return calc(t[p], qx);
61     }
62     int mid = l + r >> 1;
63     double ret = calc(t[p], qx);
64     if(qx <= mid) ret = max(ret, query(p << 1, l, mid, qx));
65     else ret = max(ret, query(p << 1 | 1, mid + 1, r, qx));
66     return ret;
67 }
68
69 void solve() {
70     cin>>n;
71     string op;
72     double s, p;
73     int t;
74     for(int i = 1; i <= n; ++i) {
75         cin>>op;
76         //cout<<i<<" "<<op<<endl;
77         if(op[0] == 'Q') {
78             cin>>t;
79             cout<<(int) (query(1, 1, sc, t) / 100)<<"\n";
80         }
81         else {

```

```

82         cin>>s>>p;
83         s -= p;
84         node newline;
85         newline.b = s;
86         newline.k = p;
87         newline.flag = 1;
88         insert(1, 1, sc, 1, sc, newline);
89     }
90 }
91 }
92
93 signed main() {
94     ios::sync_with_stdio(false);
95     cin.tie(nullptr);
96
97     int t = 1;
98     //cin>>t;
99     while(t--) {
100         solve();
101     }
102
103     return 0;
104 }

```

min25筛

定义积性函数 $f(x)$, 且 $f(p^k) = p^k(p^k - 1)$ (p 是一个质数), 求

$$\sum_{i=1}^n f(i)$$

$$g(n, j) = \sum_{i=1}^n F(i) \quad [i \in p \parallel i \text{ 的最小质因子大于第 } j \text{ 个素数}]$$

$$g(n, j) = \begin{cases} g(n, j-1) & p_j^2 > n \\ g(n, j-1) - F(p_j) \cdot \left(g\left(\left\lfloor \frac{n}{p_j} \right\rfloor, j-1\right) - \sum_{i=1}^{j-1} F(p_i) \right) & p_j^2 \leq n \end{cases}$$

$$S(n, j) = \sum_{i=1}^n f(i) \text{ [} i \text{ 的最小质因子大于第} j \text{ 个质数]}$$

$$S(n, j) = g(n, |P|) - \sum_{i=1}^{j-1} f(p_i) + \sum_{k=j+1}^{p_k^2 \leq n} \sum_{e=1}^{p_k^e \leq n} f(p_k^e) \left(S\left(\left\lfloor \frac{n}{p_k^e} \right\rfloor, k\right) + [e > 1] \right)$$

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  typedef long long ll;
5  const int N = 1e6 + 1010, mod = 1e9 + 7;
6  int n;
7
8  int ksm(int a, int b) {
9      int ret = 1;
10     while(b) {
11         if(b & 1) ret = ret * a % mod;
12         a = a * a % mod;
13         b >>= 1;
14     }
15     return ret;
16 }
17
18
19 namespace min25 {
20     int sq;
21     int g1[N], g2[N], w[N], id1[N], id2[N], tot;
22     int p[N], pr[N], cnt;
23     int sp1[N], sp2[N];
24     int inv2, inv6;
25     void init() {
26         sq = sqrt(n);
27         cnt = 0;
28         inv2 = ksm(2, mod - 2), inv6 = ksm(6, mod - 2);
29         for(int i = 2; i <= sq; ++i) {
30             if(!p[i]) {
31                 p[i] = pr[++cnt] = i;
32                 sp1[cnt] = (sp1[cnt - 1] + i) % mod;
33                 sp2[cnt] = (sp2[cnt - 1] + i*i%mod) % mod;
34             }
35             for(int j = 1; j <= cnt && i * pr[j] <= sq; ++j) {
36                 p[i * pr[j]] = pr[j];

```

```

37         if(p[i] == pr[j]) break;
38     }
39 }
40 }
41 void getG() {
42     for(int l = 1, r; l <= n; l = r + 1) {
43         r = n / (n / l);
44         int x = n / l;
45         w[++tot] = x;
46         x %= mod;
47         g1[tot] = (x * (x + 1)%mod * inv2 % mod - 1 + mod)%
mod;
48         g2[tot] = (x * (x + 1)%mod * (x * 2 + 1)% mod *
inv6%mod - 1 + mod) % mod;
49
50         w[tot] <= sq ? id1[w[tot]] = tot : id2[n / w[tot]] =
tot;
51     }
52     for(int j = 1; j <= cnt; ++j) {
53         for(int i = 1; i <= tot && pr[j] * pr[j] <= w[i];
++i) {
54             int tmp = w[i] / pr[j];
55             int p = tmp <= sq? id1[tmp] : id2[n / tmp];
56             g1[i] = (g1[i] - pr[j] * (g1[p] - sp1[j - 1] +
mod) % mod + mod) % mod;
57             g2[i] = (g2[i] - pr[j] * pr[j] %mod * (g2[p] -
sp2[j - 1] + mod) % mod + mod) % mod;
58         }
59     }
60 }
61 int getS(int i, int j) {
62     if(pr[j] >= i) return 0;
63     int p = i <= sq? id1[i] : id2[n / i];
64     int ans = ((g2[p] - g1[p] + mod) % mod - (sp2[j] -
sp1[j] + mod) % mod + mod) % mod;
65     for(int k = j + 1; pr[k] * pr[k] <= i && k <= cnt; ++k)
{
66         int pe = pr[k];
67         for(int e = 1; pe <= i; ++e, pe = pe * pr[k]) {
68             int x = pe % mod;
69             ans = (ans + x * (x - 1) % mod * (getS(i / pe,
k) + (e > 1)) % mod) % mod;

```

```
70         }
71     }
72     return ans;
73 }
74 int getans(ll n) {
75     init();
76     getG();
77
78     return gets(n, 0) + 1;
79 }
80 }
81
82 signed main() {
83     scanf("%lld", &n);
84     printf("%lld", min25::getans(n));
85     return 0;
86 }
```

$$ans_1 = \sum_{i=1}^n \varphi(i)$$

$$ans_2 = \sum_{i=1}^n \mu(i)$$

$$g(1)S(n) = \sum_{i=1}^n (f * g)(i) - \sum_{i=2}^n g(i)S(\lfloor \frac{n}{i} \rfloor)$$

1. $\mu * I = \epsilon$
2. $\varphi * I = id$
3. $\mu * id = \varphi$

```

1  #include<bits/stdc++.h>
2  #define int long long
3  using namespace std;
4  typedef long long ll;
5  const int N = 5e6 + 1010;
6  ll τ;
7  ll sphi[N], smu[N];
8  ll n;
9  int p[N], pr[N], cnt;
10 int mu[N], phi[N];
11 map<ll, ll> mpphi;
12 map<ll, ll> mpmu;
13 void init() {
14     for(int i = 2; i < N; ++i) {
15         if(!p[i]) {
16             p[i] = i;
17             pr[++cnt] = i;
18             mu[i] = -1;

```

```

19         phi[i] = i - 1;
20     }
21     for(int j = 1; j <= cnt && i * pr[j] < N; ++j) {
22         p[i * pr[j]] = pr[j];
23         if(p[i] == pr[j]) {
24             mu[i * pr[j]] = 0;
25             phi[i * pr[j]] = phi[i] * pr[j];
26             break;
27         }
28         else {
29             mu[i * pr[j]] = -mu[i];
30             phi[i * pr[j]] = phi[i] * (pr[j] - 1);
31         }
32     }
33 }
34 mu[1] = 1;
35 phi[1] = 1;
36 for(int i = 1; i < N; ++i) {
37     smu[i] = smu[i - 1] + mu[i];
38     sphi[i] = sphi[i - 1] + phi[i];
39 }
40 }
41 ll getphi(int n) {
42     if(n < N) return sphi[n];
43     if(mpphi.count(n)) return mpphi[n];
44     ll sum = (ll)(1 + n) * n / 2;
45     for(int l = 2; l <= n; ++l) {
46         int r = n / (n / l);
47         sum -= (ll)(r - l + 1) * getphi(n / l);
48         l = r;
49     }
50     return mpphi[n] = sum;
51 }
52 ll getmu(int n) {
53     if(n < N) return smu[n];
54     if(mpmu.count(n)) return mpmu[n];
55     ll sum = 1;
56     for(int l = 2; l <= n; ++l) {
57         int r = n / (n / l);
58         sum -= (ll)(r - l + 1) * getmu(n / l);
59         l = r;
60     }

```

```

61     return mpmu[n] = sum;
62 }
63 signed main() {
64     init();
65     scanf("%lld", &T);
66     while(T--) {
67         scanf("%lld", &n);
68         printf("%lld %lld\n", getphi(n), getmu(n));
69     }
70     return 0;
71 }

```

求莫比乌斯函数值平方前缀和

令 $S(n) = \sum_{i=1}^n (\mu(i))^2$ ($n < 2^{31}$)

这个题和之前的就有所不同了。我们令 $f(i) = \mu(i)^2$ ，但是要找方便计算的 g 才行。

我们选取函数 $g(x) = [x = k^2 \quad k \in N^+]$ 。我们来算一下 g 和 f 的狄利克雷卷积。我们会发现 $f * g = 1$ ！简要证明一下：

$(f * g)(n) = \sum_{d|n} g(d) \cdot f(\frac{n}{d})$ 。首先分类讨论，如果 d 是一个完全平方数，只有当 d 是 n 的所有约数中最大的完全平方数时，乘积为 1，否则 $\mu(\frac{n}{d}) = 0$ 。如果 d 不是完全平方数，那么 $g(d) = 0$ 。得证。

那我们的计算就简化了不少。

$$\begin{aligned}
 g(1) \cdot S(n) &= \sum_{i=1}^n (f * g)(i) - \sum_{i=2}^n g(i) \cdot S(\lfloor \frac{n}{i} \rfloor) \\
 1 \cdot S(n) &= \sum_{i=1}^n 1 - \sum_{i=2}^{\sqrt{n}} 1 \cdot S(\lfloor \frac{n}{i^2} \rfloor) \\
 S(n) &= n - \sum_{i=2}^{\sqrt{n}} S(\lfloor \frac{n}{i^2} \rfloor)
 \end{aligned}$$


```

1  const int N = 1 << 17 | 1;
2  int n, m;
3  modint A[N], B[N], a[N], b[N];
4
5  inline void in() {
6      for (int i = 0; i < n; i++) a[i] = A[i], b[i] = B[i];
7  }
8
9  inline void get() {
10     for (int i = 0; i < n; i++) a[i] *= b[i];
11 }
12
13 inline void out() {
14     for (int i = 0; i < n; i++) print(a[i], " \n"[i==n-1]);
15 }
16
17 inline void OR(modint *f, modint x = 1) {
18     for (int o = 2, k = 1; o <= n; o <= 1, k <= 1)
19         for (int i = 0; i < n; i += o)
20             for (int j = 0; j < k; j++)
21                 f[i+j+k] += f[i+j] * x;
22 }
23
24 inline void AND(modint *f, modint x = 1) {
25     for (int o = 2, k = 1; o <= n; o <= 1, k <= 1)
26         for (int i = 0; i < n; i += o)
27             for (int j = 0; j < k; j++)
28                 f[i+j] += f[i+j+k] * x;
29 }
30
31 inline void XOR(modint *f, modint x = 1) {
32     for (int o = 2, k = 1; o <= n; o <= 1, k <= 1)
33         for (int i = 0; i < n; i += o)
34             for (int j = 0; j < k; j++)
35                 f[i+j] += f[i+j+k],
36                 f[i+j+k] = f[i+j] - f[i+j+k] - f[i+j+k],
37                 f[i+j] *= x, f[i+j+k] *= x;
38 }
39

```

```
40 int main() {
41     rd(m), n = 1 << m;
42     for (int i = 0; i < n; i++) rd(A[i]);
43     for (int i = 0; i < n; i++) rd(B[i]);
44     in(), OR(a), OR(b), get(), OR(a, P - 1), out();
45     in(), AND(a), AND(b), get(), AND(a, P - 1), out();
46     in(), XOR(a), XOR(b), get(), XOR(a, (modint)1 / 2), out();
47     return 0;
48 }
```

卢卡斯定理

```
1 ll lucas(int n,int m,int p){
2     if(m==0) return 1;
3     return lucas(n/p,m/p,p)*C(n%p,m%p)%p;
4 }
```

扩展卢卡斯定理

```
1 #include<bits/stdc++.h>
2 using namespace std;
3 #define ll long long
4 const int N = 2e6+100;
5 ll a[N], p[N], pe[N];
6 ll fac[N];
7 int cnt = 0;
8 ll T, n, m;
9 ll mod;
10 int now;
11 void exgcd(ll a, ll b, ll &x, ll &y){
12     if(!b){
13         x = 1;
14         y = 0;
15         return;
16     }
17     exgcd(b, a%b, y, x);
18     y -= (a / b) * x;
19 }
20 ll qpow(ll a,ll b,ll p){
21     ll c=1;
22     while(b){
23         if(b&1)c=(c*a)%p;
24         a=(a*a)%p;
25         b>>=1;
26     }
27     return c;
28 }
29 ll inv(ll a,ll p){
30     ll x, y;
31     exgcd(a, p, x, y);
32     x %= p;
```

```

33     if(x < 0) x += p;
34     return x;
35 }
36
37 void init() {
38     int pp = mod;
39     for(int i = 2; i * i <= pp; ++i){
40         if(pp%i) continue;
41         ll tmp=1;
42         p[++cnt] = i;
43         pe[cnt] = 1;
44         while(pp % i == 0){
45             pp/=i;
46             pe[cnt] *= i;
47         }
48     }
49     if(pp > 1) {
50         p[++cnt] = pp;
51         pe[cnt] = pp;
52     }
53 }
54
55
56
57 ll facdiv(ll n, ll p, ll pk){
58     if(!n) return 1;
59     ll ans=1;
60     for(ll i = 1; i < pk; ++i){
61         if(i % p) ans = (ans * i) % pk;
62     }
63     ans = qpow(ans, n / pk, pk);
64     for(ll i = 1; i <= n % pk; ++i){
65         if(i % p) ans = (ans * i) % pk;
66     }
67     return ans * facdiv(n / p, p, pk) % pk;
68 }
69 ll c(ll n, ll m, ll p, ll pk){
70     if(n < m) return 0;
71     ll f1 = facdiv(n, p, pk), f2 = facdiv(m, p, pk), f3 =
facdiv(n - m, p, pk), cnt=0;
72     ll t1 = n, t2 = m, t3 = n - m;
73     for(; t1; t1/=p) cnt += t1/p;

```

```

74     for(; t2; t2/=p) cnt -= t2/p;
75     for(; t3; t3/=p) cnt -= t3/p;
76     return ((f1*inv(f2,pk) %
pk)*inv(f3,pk)%pk)*qpow(p,cnt,pk)%pk;
77 }
78
79
80 ll exlucas(ll n,ll m,int pp){
81     ll x;
82     ll ret = 0;
83     for(int i = 1; i <= cnt; ++i) {
84         x = C(n, m, p[i], pe[i]);
85         ret = (ret + ((mod / pe[i] * x) % mod * inv(mod /
pe[i], pe[i]) % mod))%mod;
86     }
87     return ret;
88 }
89 int main(){
90     //scanf("%lld%lld", &mod, &T);
91     //T = 1;
92     scanf("%lld%lld%lld",&n,&m, &mod);
93     init();
94     printf("%lld\n",exlucas(n,m,mod));
95
96     // while(T--) {
97     //     scanf("%lld%lld",&n,&m);
98     //     printf("%lld\n",exlucas(n,m,mod));
99     // }
100     return 0;
101 }

```

第一类斯特林数-行

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  #define int long long
4  typedef long long ll;
5  const ll mod=167772161;
6  ll G=3,invG;
7  const int N=1200000;
8  ll ksm(ll b,int n){

```

```

9     ll res=1;
10    while(n){
11        if(n&1) res=res*b%mod;
12        b=b*b%mod; n>>=1;
13    }
14    return res;
15 }
16 int read(){
17     int x=0;char ch=getchar();
18     while(!isdigit(ch))ch=getchar();
19     while(isdigit(ch)) x=(x*10+(ch-'0'))%mod,ch=getchar();
20     return x;
21 }
22 int tr[N];
23 void NTT(ll *f,int n,int fl){
24     for(int i=0;i<n;++i)
25         if(i<tr[i]) swap(f[i],f[tr[i]]);
26     for(int p=2;p<=n;p<<=1){
27         int len=(p>>1);
28         ll w=ksm((fl==0)?G:invG,(mod-1)/p);
29         for(int st=0;st<n;st+=p){
30             ll buf=1,tmp;
31             for(int i=st;i<st+len;++i)
32                 tmp=buf*f[i+len]%mod,
33                 f[i+len]=(f[i]-tmp+mod)%mod,
34                 f[i]=(f[i]+tmp)%mod,
35                 buf=buf*w%mod;
36         }
37     }
38     if(fl==1){
39         ll invN=ksm(n,mod-2);
40         for(int i=0;i<n;++i)
41             f[i]=(f[i]*invN)%mod;
42     }
43 }
44 void Mul(ll *f,ll *g,int n,int m){
45     m+=n;n=1;
46     while(n<m) n<<=1;
47     for(int i=0;i<n;++i)
48         tr[i]=(tr[i>>1]>>1)|((i&1)?(n>>1):0);
49     NTT(f,n,0);
50     NTT(g,n,0);

```

```

51     for(int i=0;i<n;++i) f[i]=f[i]*g[i]%mod;
52     NTT(f,n,1);
53 }
54 ll inv[N],fac[N];
55 ll w[N],a[N],b[N],g[N];
56 void Solve(ll *f,int m){
57     if(m==1) return f[1]=1,void(0);
58     if(m&1){
59         solve(f,m-1);
60         for(int i=m;i>=1;--i)
61             f[i]=(f[i-1]+f[i]*(m-1)%mod)%mod;
62         f[0]=f[0]*(m-1)%mod;
63     }
64     else{
65         int n=m/2;ll res=1;
66         solve(f,n);
67         for(int i=0;i<=n;++i)
68             a[i]=f[i]*fac[i]%mod,b[i]=res*inv[i]%mod,res=res*n%mod;
69         reverse(a,a+n+1);
70         Mul(a,b,n+1,n+1);
71         for(int i=0;i<=n;++i)
72             g[i]=inv[i]*a[n-i]%mod;
73         Mul(f,g,n+1,n+1);
74         int limit=1;
75         while(limit<(n+1)<<1) limit<<=1;
76         for(int i=n+1;i<limit;++i) a[i]=b[i]=g[i]=0;
77         for(int i=m+1;i<limit;++i) f[i]=0;
78     }
79 }
80 ll f[N];
81 void init(int n){
82     fac[0]=1;
83     for(int i=1;i<=n;++i)
84         fac[i]=1ll*fac[i-1]*i%mod;
85     inv[n]=ksm(fac[n],mod-2);
86     for(int i=n-1;i>=0;--i)
87         inv[i]=1ll*inv[i+1]*(i+1)%mod;
88 }
89 signed main(){
90     invG=ksm(G,mod-2);
91     int n,k=0;

```

```

92     cin>>n;
93     init(n+n);
94     solve(f,n);
95     for(int i=0;i<=n;++i)
96         printf("%lld ",f[i]);
97     return 0;
98 }

```

卡特兰数

卡特兰数，一个特殊的数列。通项公式为：

$$Cat_n = \frac{C_{2n}^n}{n+1}$$

从0开始的前几项为：1, 1, 2, 5, 14, 42, 132, ...，所以有的题可以直接打个表看看（比如[这个](#)）

然后它是怎么推出来的，最主要的就是从(0, 0)到(n, n)不穿过直线 $y = x$ 的路径计数（不想上图了，可以手画一个）。首先我们随便走的走法就是2n步里面选n步向上剩下n步向右，就是 C_{2n}^n 。

然后减去不合法的方案数。我们发现，如果穿过直线 $y = x$ ，那必然接触直线 $y = x + 1$ 。然后我们把第一个接触点之后向右和向上的走法反转，那么它就会走到(n - 1, n + 1)，走法数显然是 C_{2n}^{n+1} 。于是一个公式就是

$$Cat_n = C_{2n}^n - C_{2n}^{n+1}$$

还有一些其他的公式：

$$Cat_n = \frac{Cat_{n-1}(4n-2)}{n+1}$$

$$Cat_n = \sum_{i=1}^n Cat_{i-1} Cat_{n-i} (n \geq 2)$$

最后是一些常用的卡特兰数模型：

1. 一个01串，n个0n个1。使任意前缀中0的个数不小于1的个数的方案数为 Cat_n 。
2. n个点的有标号二叉树的个数为 Cat_n 。
3. 一个栈的进栈序列为1, 2, ..., n, 则不同的出栈序列个数为 Cat_n 。
4. 圆上2n个点，用n条线段成对连接，不相交的方案数为 Cat_n 。
5. 将一个凸多边形剖分成n个三角形的方案数为 Cat_n 。

prufer序列

一个长度为n-2的Prufer序列，唯一对应一棵n个点固定形态的无根树。

性质：

1. prufer序列中，点u出现的次数，等于点u在树中的度数-1
2. n个点的无根树，唯一对应长度为n-2的prufer序列，序列每个数都在1到n的范围内。

3. Cayley定理: n 个点的无向完全图的生成树的计数: n^{n-2} , 即 n 个点的有标号无根树的计数

4. n 个节点的度依次为 d_1, d_2, \dots, d_n 的无根树共有 $\frac{(n-2)!}{\prod_{i=1}^n (d_i - 1)!}$ 个, 因为此时
Prufer编码中的数字 i 恰好出现 $d_i - 1$ 次, $(n-2)!$ 是总排列数

5. n 个点的有标号有根树的计数: $n^{n-2} * n = n^{n-1}$

矩阵树定理

给出一个无向无权图, 设 A 为邻接矩阵, D 为度数矩阵($D[i][i]$ = 节点 i 的度数, 其他的无值)。

则基尔霍夫(Kirchhoff)矩阵即为: $K = D - A$

然后令 K' 为 K 去掉第 k 行与第 k 列(k 任意)的结果($n - 1$ 阶主子式),

$\det(K')$ 即为该图的生成树个数。

• 有向扩展

前面都是无向图, 神奇的是有向图的情况也是可以做的。

(邻接矩阵 A 的意义同有向图邻接矩阵)

那么现在的矩阵 D 就要变一下了。

若 $D[i][i] = \sum_{j=1}^n A[j][i]$, 即到该点的边权总和(入)。

此时求的就是**外向树**(从根向外)

若 $D[i][i] = \sum_{j=1}^n A[i][j]$, 即从该点出发的边权总和(出)。

此时求的就是**内向树**(从外向根)

(如果考场上不小心忘掉了, 可以手玩小样例)

(同样可以加权!)

此外, 既然是有向的, 那么就需要**指定根**。

前面提过要任意去掉第 k 行与第 k 列, 是因为无向图所以不用在意谁为根。

在有向树的时候需要理解为指定根, 结论是: 去掉哪一行就是那一个元素为根。

二项式反演 (3个形式)

$$f(n) = \sum_{i=0}^n (-1)^i \binom{n}{i} g(i)$$

$$g(n) = \sum_{i=0}^n (-1)^i \binom{n}{i} f(i)$$

$$f(n) = \sum_{i=0}^n \binom{n}{i} h(i) \Leftrightarrow \frac{h(n)}{(-1)^n} = \sum_{i=0}^n (-1)^i \binom{n}{i} f(i)$$

$$f(n) = \sum_{i=n}^m \binom{i}{n} g(i) \Leftrightarrow g(n) = \sum_{i=n}^m (-1)^{i-n} \binom{i}{n} f(i)$$

第一类斯特林数

n 个不同元素构成 m 个圆的排列方案数

$$s_u(n, m) = s_u(n-1, m-1) + s_u(n-1, m) * (n-1)$$

第一类斯特林数列

思路

首先，我们可以对 $k = 1$ 的情况构造指数级生成函数，即：

$$S(x) = \sum_{i=0}^n (i-1)! \frac{x^i}{i!}$$

因为很显然 $\begin{bmatrix} n \\ 1 \end{bmatrix} = (n-1)!$ 。

那么，我们就可以得到，对于 k 为任意数情况的指数级生成函数就是：

$$\frac{S(x)^k}{k!}$$

除以 $k!$ 的主要原因是我们用指数级生成函数的环排列其实是有顺序。就比如 $[2, 3, 1][1, 2]$ 和 $[1, 2][2, 3, 1]$ 是等价的，但是我们算重了。

上面这个式子如果要美观一点就是：

$$\frac{(\ln \frac{1}{1-x})^k}{k!}$$

这个可以通过泰勒展开得到。

不过不管用哪种表达方式都可以，直接多项式快速幂就好了。不过我的 $\Theta(n \log^2 n)$ 的朴素版似乎卡不过去。看来需要练习卡常技巧了，这里就只给出 $\Theta(n \log n)$ 的代码。

第二类斯特林数

n 个不同元素构成 m 个集合的排列方案数

$$S(n, m) = S(n-1, m-1) + m \times S(n-1, m)$$

$$s_u(n, m) = s_u(n-1, m-1) + s_u(n-1, m) * (n-1)$$

$$S(n, m) = \frac{1}{m!} \sum_{k=0}^m (-1)^k \frac{m!}{k!(m-k)!} (m-k)^n$$

$$S(n, m) = \frac{1}{m!} \sum_{k=0}^m m! \frac{(-1)^k}{k!} \frac{(m-k)^n}{(m-k)!}$$

$$S(n, m) = \sum_{k=0}^m \frac{(-1)^k}{k!} \frac{(m-k)^n}{(m-k)!}$$

第二类斯特林数列

$$\sum_{n=k}^{\infty} \left\{ \begin{matrix} n \\ k \end{matrix} \right\} \frac{x^n}{n!} = \frac{(e^x - 1)^k}{k!}$$

斯特林反演:
$$f(n) = \sum_{k=0}^n \left\{ \begin{matrix} n \\ k \end{matrix} \right\} g(k) \iff g(n) = \sum_{k=0}^n (-1)^{n-k} \left[\begin{matrix} n \\ k \end{matrix} \right] f(k)$$

下降幂

下降幂:

$$x^{\underline{m}} = x(x-1) \cdots (x-m+1) = m! \binom{x}{m} = \frac{x!}{(x-m)!}$$

下降幂的差分:

$$(x+1)^{\underline{m}} - x^{\underline{m}} = mx^{\underline{m-1}}$$

下降幂的定和式:

$$\sum_{a \leq x < b} x^{\underline{m}} = \frac{b^{\underline{m+1}} - a^{\underline{m+1}}}{m+1}$$

贝尔数

$$\exp(e^x - 1)$$

贝尔数 B_n 是基数（元素个数）为 n 的集合的划分方法的数目。集合 S 的一个划分是定义为 S 的两两不相交的非空子集的族，它们的并是 S 。

正文

首先根据贝尔数的定义，有

$$B_n = \sum_{m=0}^n S(n, m)$$

其中 $S(n, m)$ 是第二类斯特林数。

那么再由第二类斯特林数的展开式可得

$$\begin{aligned} \text{原式} &= \sum_{m=0}^n \frac{1}{m!} \sum_{k=0}^m (-1)^k C(m, k) (m-k)^n \\ &= \sum_{m=0}^n \sum_{k=0}^m \frac{(-1)^k}{k!} \frac{(m-k)^n}{(m-k)!} \end{aligned}$$

这样子，设 $A_i = \frac{(-1)^i}{i!}$ ， $B_i = \frac{i^n}{i!}$ ，这样子就是

$$\text{原式} = \sum_{m=0}^n \sum_{k=0}^m A_k B_{m-k}$$

可以 NTT ，但是太麻烦，我们注意到对于 A_i 这一项，它只会与 $B_0, B_1, B_2 \dots B_{n-i}$ 相乘，就是一个前缀和的形式，所以 A_i 这一项的贡献就算了出来，这样的话，预处理 A, B 以及其前缀和，然后 for 一遍，把每一项 A_i 的贡献算出来加上去就可以了，这样子是 $O(n \log n)$ 的（要算快速幂）。

$$B_{n+1} = \sum_{i=0}^n \binom{n}{i} B_i$$

$$B_{p+n} \equiv B_n + B_{n+1} \pmod{p}$$

$$B_{p^m+n} \equiv mB_n + B_{n+1} \pmod{p}$$

MENU

0x01 兰道定理

0x02 寻找强连通分量

0x03 哈密顿通路

0x04 哈密顿回路

竞赛图是把一个完全图的边定向后得到的有向图，所以也是一个 n 个点 $\binom{n}{2}$ 条边的无自环重边的有向图。

竞赛图有许多优美的性质和定理，并且多半都和强连通分量有关系。

0x01 兰道定理

对于一个出度序列 $s_1 \dots s_n$ ，它是合法的（存在一个竞赛图出度满足这个序列），当且仅当：

$$\forall 1 \leq k \leq n, \sum_{i=1}^k s_i \geq \binom{k}{2}$$

且在 $k = n$ 时取等。注意到把以上结论中的出度换成入度也是对的。

必要性比较显然，因为前 k 个点内部的边已经刚好 $\binom{k}{2}$ 条了。

充分性不太会证，好像是可以构造出来的。

0x02 寻找强连通分量

首先我们可以手模一下竞赛图缩点之后的结构，显然入度为 0 和出度为 0 的 scc 都只能有一个，所以这应该是一个链的结构。

可以发现，拓扑序小的 scc 里面的点的出度一定严格大于拓扑序大的 scc 里面的点的出度。

所以所有点按照出度大小从小到大排序之后，这个图的每个 scc 都是序列上的一个区间。

我们可以找到第一个 k 使得 $\sum_{i=1}^k s_i = \binom{k}{2}$ ，发现这个序列上的前 k 个点就是原图拓扑序最大的 scc。因为首先肯定有 $\sum_{i=1}^k s_i \geq \binom{k}{2}$ ，如果等于的话说明后面的点与这 k 个点的边都是连向这 k 个点的，这是前 k 个点组成一个强连通分量的必要条件。并且前面不存在取等的地方，说明前面绝对没有强连通分量，所以这 k 个点恰好组成了一个强连通分量。

简单地拓展一下可以发现，找到所有的 k 使得兰道定理的不等式取等，这些 k 就是所有强连通分量的分界点。

0x03 哈密顿通路

MENU

0x01 兰道定理

0x02 寻找强连通分量

0x03 哈密顿通路

0x04 哈密顿回路

简单地拓展一下可以发现，找到所有的 k 使得兰道定理的不等式取等，这些 k 就是所有强连通分量的分界点。

0x03 哈密顿通路

一个结论是：任何竞赛图都存在哈密顿通路。

我们可以尝试构造出一组解。

考虑增量构造，从空图开始加点，现在加入点 x ，设前面的点形成的哈密顿通路的起点为 S ，终点为 T 。

- 如果存在 $x \rightarrow S$ 或者 $T \rightarrow x$ 的边，则直接将 x 在端点处加入通路即可。
- 否则此时一定存在 $S \rightarrow x$ 和 $x \rightarrow T$ ，那么这条通路上一定存在两个相邻的点 u, v 使得 $u \rightarrow x, x \rightarrow v$ ，在 u, v 之间插入 x 即可。

这样就完成了证明，并且我们找到了一个 $O(n^2)$ 构造哈密顿通路的算法。

0x04 哈密顿回路

一个结论是：任何强连通的竞赛图都存在哈密顿回路。

我们可以尝试构造一组解。

首先我们先找出一条这个竞赛图的哈密顿通路，不妨认为这个通路是 $1 \rightarrow 2 \rightarrow 3 \dots \rightarrow n-1 \rightarrow n$ 。

因为强连通，所以肯定存在一个点使得 $t \rightarrow 1$ 。我们可以把 $1 \rightarrow 2 \rightarrow \dots \rightarrow t \rightarrow 1$ 作为初始的回路，然后考虑增量构造。

现在加入点 x ，且前 $x-1$ 个点已经构造出了一条回路。

- 如果前 $x-1$ 个点中存在一个 y 使得 $x \rightarrow y$ ，注意到 $x-1 \rightarrow x$ 成立，所以我们一定可以找到回路上相邻的两个点 u, v 使得 $u \rightarrow x, x \rightarrow v$ ，把 x 插入到 u, v 间即可。
- 否则前 $x-1$ 个点中的每一个点都连向了 x ，又因为强连通，后面肯定存在一个 y 连有到前 $x-1$ 个点的反向边，找到这个 y ，设它的反向边连向了 t 。设 t 在回路上的前一个点是 r ，因为有 $r \rightarrow x$ 的边，我们可以像 $r \rightarrow x \rightarrow x+1 \dots \rightarrow y \rightarrow t$ 这样把 $[x, y]$ 这一段的点都加入回路。

这样就完成了证明，并且我们找到了一个 $O(n^2)$ 构造哈密顿回路的算法。

```
1 void init()
2 {
3     vis.clear();
4     b.clear();
5     a[1][1]=1;
6     for(int i=2;i<=p;++i) for(int j=1;j<=i;++j) {
7         if(j==1) a[i][j]=a[i-1][i-1];
8         else {
9             a[i][j]=(a[i][j-1]+a[i-1][j-1])%p;
10        }
11    }
12 }
13
14 int get_ans(int x)
15 {
```

```
16     if(x<=p) {
17         return a[x][0];
18     }
19     if(vis[x]) return b[x];
20     vis[x]=1;
21     return b[x]=(get_ans(x-p)+get_ans(x-p+1))%p;
22
```