一、 dp 相关算法

1. 数位 dp

```cpp
#include<bits/stdc++.h>
#define B(x) (1<<(x))
using namespace std;
const int maxn = 20;
const int maxf = 20000;

int num[maxn], l;
int fa, pow2[maxn];
int dp[maxn][maxf];

int f(int x) {
    int l2=0;
    int rtn = 0;
    while(x) {
        rtn += (x%10)*B(l2++);
        x/=10;
    }
    return rtn;
}

int dfs(int len,int sum,int lim) {
    if(len==-1) return sum >= 0;
    if(sum < 0) return 0;
    int ans = dp[len][sum];
    if(ans!=-1 && !lim)
        return ans;
    ans = 0;
    int top=lim?num[len]:9;
    int nsum,nlim;
    for(int i=0;i<=top;i++)
        ans += dfs(len-1, sum-i*B(len), lim&&i==top);
    if(!lim) dp[len][sum] = ans;
    return ans;
}

int solve(int x) {
    l = 0;
    while(x)num[l++]=x%10,x/=10;
    return dfs(l-1,fa,1);
```

```
}

int main() {
    int T, kase=1;
    scanf("%d",&T);
    memset(dp,-1,sizeof(dp));
    while(T--) {
        int a, b;
        scanf("%d%d",&a,&b);
        fa = f(a);
        printf("Case #%d: %d\n", kase++, solve(b));
    }
}
```

2. 斜率 dp

```
#include<bits/stdc++.h>
//#define __LOCAL_TEST__
#define inf 0x3f3f3f3f3f3f
using namespace std;
typedef long long ll;

void debug() { cout << endl; }
template<typename T,typename ...R>
    void debug(T t, R ...r) { cout << "[" << t << "] "; debug(r...); }

template<class T> inline bool Scanf(T& t) {
    char c=getchar();t=0;
    int f = 1;
    while(c!='-'&&c!=EOF&&(c<'0'||c>'9'))c=getchar();
    if(c==EOF) return 0;
    if(c=='-')f=-1,c=getchar();
    while(c>='0'&&c<='9')t=t*10+(T)(c-'0'),c=getchar();
    t = f*t; return 1;
}

const int maxn = 500000 + 5;

ll dp[maxn], sum[maxn];
int que[maxn<<1];

ll dpX(int i)
{
```

```
    return sum[i];
}

ll dpY(int i)
{
    return dp[i] + sum[i]*sum[i];
}

int main()
{
#ifdef __LOCAL_TEST__
    freopen("1.in", "r", stdin);
#endif // __LOCAL_TEST__
    int n, m;
    while(scanf("%d%d", &n, &m)!=EOF) {

        sum[0] = 0; dp[0] = 0;
        for(int i=1;i<=n;++i) {
            ll x; Scanf(x);
            sum[i] = sum[i-1]+x;
        }

        int h = 1, t = 0;
        que[++t] = 0;
        for(int i=1;i<=n;i++) {
            while(h<t && (dpY(que[h])-dpY(que[h+1]))>=2*sum[i]*(dpX(que[h])-dpX(que[h+1])))
                h ++;
            dp[i] = dp[que[h]] + (sum[i]-sum[que[h]])*(sum[i]-sum[que[h]]) + m;
            while(h<t && (dpY(que[t])-dpY(i))*(dpX(que[t-1])-dpX(que[t]))<=(dpY(que[t-1])-dpY(que[t]))*(dpX(que[t])-dpX(i)))
                t --;
            que[++t] = i;
        }
        printf("%lld\n", dp[n]);
    }
    return 0;
}
```

3.　二维斜率优化 dp
/*分析:
首先对于斜率 dp 我有个总结:
斜率 dp 一般应用于连续的一段或几段求最值
既 1~k,k+1~j,j+1~...这样分段而不能跳开来求
仅仅有连续段才干用单调队列维护最值然后
dp[i]=dp[j]+(j+1~i)的值。

对于本题:
题目要求 m 个子数组的最值。而子数组中的元素不一定是原数组连续的
所以肯定不能直接用斜率优化，经过分析能够发现先进行从小到大排序
然后连续的 m 段最值就是能够求最值了。

所以：先对原数组进行从小到大排序
dp[i][j]表示以 i 结尾的 j 段的最值
从 k+1~i 作为一段
则：dp[i][j]=dp[k][j-1]+(s[i]-s[k+1])^2
如今就是怎样求到这个 k 使得 dp[i][j]最小
如果 k2<=k1<i
若:dp[k1][j-1]+(s[i]-s[k1+1])^2 <= dp[k2][j-1]+(s[i]-s[k2+1])^2
=>dp[k1][j-1]+s[k1+1]^2 - (dp[k2][j-1]+s[k2+1]^2) / (2s[k1+1]-2s[k2+1]) <= s[i]
所以:
y1 = dp[k1][j-1]+s[k1+1]^2
x1 = 2s[k1+1]
y2 = dp[k2][j-1]+s[k2+1]^2
x2 = 2s[k2+1]

=>(y1 - y2)/(x1 - x2) <= i
单调队列维护下凸折线
*/
```
#include <iostream>
#include <cstdio>
#include <cstdlib>
#include <cstring>
#include <string>
#include <queue>
#include <algorithm>
#include <map>
#include <cmath>
#include <iomanip>
#include <limits.h>
#define INF 99999999
typedef long long LL;
using namespace std;
```

```
const int MAX = 10000+10;
int n,m,index;
int q[MAX];
int s[MAX],dp[2][MAX];//采用滚动数组

int GetY(int k1,int k2){
	return dp[index^1][k1]+s[k1+1]*s[k1+1] - (dp[index^1][k2]+s[k2+1]*s[k2+1]);
}

int GetX(int k1,int k2){
	return 2*(s[k1+1]-s[k2+1]);
}

int DP(){
	int head=0,tail=1;
	index=0;
	for(int i=1;i<=n;++i)dp[index][i]=INF;//初始化
	//dp[index][0]=0;
	for(int i=1;i<=m;++i){
		index=index^1;
		head=tail=0;
		q[tail++]=0;
		for(int j=1;j<=n;++j){
			//dp[index^1][0]=(i-1)*(s[j]-s[1])*(s[j]-s[1]);
			while(head+1<tail && GetY(q[head+1],q[head]) <= GetX(q[head+1],q[head])*s[j])++head;
			while(head+1<tail && GetY(j,q[tail-1])*GetX(q[tail-1],q[tail-2]) <= GetY(q[tail-1],q[tail-2])*GetX(j,q[tail-1]))--tail;
			q[tail++]=j;
			int k=q[head];
			dp[index][j]=dp[index^1][k]+(s[j]-s[k+1])*(s[j]-s[k+1]);
		}
	}
	return dp[index][n];
}

int main(){
	int t,num=0;
	scanf("%d",&t);
	while(t--){
		scanf("%d%d",&n,&m);
		for(int i=1;i<=n;++i)scanf("%d",s+i);
		sort(s+1,s+1+n);
```

```
        printf("Case %d: %d\n",++num,DP());
    }
    return 0;
}
```

4. 四边形不等式优化 dp

```
#include <fstream>
#include <iostream>
#include <cstdio>
#include <cstring>
#include <cstdlib>
#include <cmath>
using namespace std;

const int N=205;
const int INF=0x7fffffff;
int n;
int a[N],sum[N],dp[N][N],s[N][N];

void f();

int main(){
    //freopen("D:\\input.in","r",stdin);
    while(~scanf("%d",&n)){
        sum[0]=0;
        for(int i=1;i<=n;i++){
            scanf("%d",&a[i]);
            sum[i]=sum[i-1]+a[i];
        }
        f();
        printf("%d\n",dp[1][n]);
    }
    return 0;
}
void f(){
    for(int i=1;i<=n;i++) dp[i][i]=0,s[i][i]=i;
    for(int r=1;r<n;r++){
        for(int i=1;i<n;i++){
            int j=i+r;
```

```
            if(j>n) break;
            dp[i][j]=INF;
            for(int k=s[i][j-1];k<=s[i+1][j];k++){
                if(dp[i][j]>dp[i][k]+dp[k+1][j]){
                    dp[i][j]=dp[i][k]+dp[k+1][j];
                    s[i][j]=k;
                }
            }
            dp[i][j]+=sum[j]-sum[i-1];
        }
    }
}
```

5. SOSdp

```
#include <bits/stdc++.h>
using namespace std;

int sum[100];

int main() {
   int n=5;
   sum[1]=1;sum[3]=3;
   for (int i = 0; i < n; ++ i) {
      for (int s = 0; s < (1<<n); ++ s)
         if ((s >> i) & 1) sum[s] += sum[s ^ (1 << i)];
   }
   int x;
   while (cin >> x) {
      cout << sum[x] << endl;
   }
}
```

1. 莫队

```
#include <bits/stdc++.h>
using namespace std;

const int N = 510000;

int a[N], num[N];
int blong[N];
int ans[N];
int cnt, cot;
```

```
struct Q
{
  int l, r, id;
  bool operator < (const Q& ot) const {
    if (blong[l] ^ blong[ot.l]) return blong[l] < blong[ot.l];
    return (blong[l] & 1) ? r < ot.r : r > ot.r;
  }
} p[N];

void add(int x)
{
  if (num[a[x]] == 2) cot --;
  num[a[x]] ++;
  if (num[a[x]] == 2) cot ++;
}

void del(int x)
{
  if (num[a[x]] == 2) cot --;
  num[a[x]] --;
  if (num[a[x]] == 2) cot ++;
}

int main()
{
  int n, q;
  scanf("%d%d", &n, &q);
  for (int i = 0; i < n; ++ i) {
    scanf("%d", a + i);
    num[i] = a[i];
  }
  sort(num, num + n);
  cnt = unique(num, num + n) - num;
  int sqt = sqrt(n);
  int bid = 0;
  for (int i = 0; i < n; ++ i) {
    a[i] = lower_bound(num, num + cnt, a[i]) - num;
    if (i % sqt == 0) bid ++;
    blong[i] = bid;
  }
  for (int i = 0; i < q; ++ i) {
    scanf("%d%d", &p[i].l, &p[i].r);
    p[i].l --;
```

```
    p[i].r --;
    p[i].id = i;
  }
  sort(p, p + q);
  memset(num, 0, sizeof(num));
  int l = 0, r = 0;
  cot = 0;
  num[a[0]] = 1;
  for (int i = 0; i < q; ++ i) {
    while (l < p[i].l) del(l ++);
    while (l > p[i].l) add(-- l);
    while (r < p[i].r) add(++ r);
    while (r > p[i].r) del(r --);
    ans[p[i].id] = cot;
  }
  for (int i = 0; i < q; ++ i) {
    printf("%d\n", ans[i]);
  }
  return 0;
}
```

二、 数学相关

1. CHRT

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

namespace CHRT
{
  void exp_gcd(long long a, long long b, long long &g, long long &x, long long &y)
  {
    if(b == 0) {
      x = 1; y = 0; g = a; return ;
    }
    exp_gcd(b, a % b, g, y, x); y -= x * (a / b);
  }
  long long cal(std::vector<long long> &a, std::vector<long long> &m)
  {
    int siz = a.size();
    long long a1 = a[0], m1 = m[0];
    long long mg = m[0];
    for (int i = 1; i < siz; ++ i) {
```

```
            mg = mg / std::__gcd(mg, m[i]) * m[i];
            long long a2 = a[i];
            long long m2 = m[i];
            long long x, y, g;
            exp_gcd(m1, m2, g, x, y);
            long long d = a2 - a1;
            if(d % g != 0) return -1;
            long long ksm = d / g * x;
            long long tsm = std::abs(m2 / g);
            ksm = (ksm % tsm + tsm) % tsm;
            a1 = a1 + m1 * ksm;
            m1 = m1 / g * m2;
        }
        return a1;
        //return (a1 == 0 ? a1 + mg : a1);  要求非 0
    }
}

int main()
{
    int T, kase = 1;
    std::cin >> T;
    while (T --) {
        std::vector<long long> a, b;
        int n;
        std::cin >> n;
        a.resize(n);
        b.resize(n);
        for (int i = 0; i < n; ++ i) {
            std::cin >> b[i];
        }
        for (int i = 0; i < n; ++ i) {
            std::cin >> a[i];
        }
        std::cout << "Case " << kase ++ << ": " << CHRT::cal(a, b) << std::endl;
    }
    return 0;
}
```

2. FFT

```
#include<iostream>
#include<cstdio>
#include<cstring>
#include<algorithm>
```

```
#include<complex>
#include<cmath>
#define pi acos(-1)
using namespace std;
const int MAXN=131072+5;
typedef complex<double> com;
int n,m,L;
com a[MAXN],b[MAXN];
int c[MAXN],Rev[MAXN];

void get_bit(){for (n=1,L=0;n<m;n<<=1) L++;}
void get_Rtable(){for (int i=0;i<n;i++) Rev[i]=(Rev[i>>1]>>1)|((i&1)<<(L-1));}
void multi(com* a,com* b){for (int i=0;i<n;i++) a[i]*=b[i];}

void FFT(com* a,int flag)
{
	for (int i=0;i<n;i++)if(i<Rev[i])swap(a[i],a[Rev[i]]); //利用逆序表，快速求逆序
	for (int i=1;i<n;i<<=1)
	{
		com wn(cos(2*pi/(i*2)),flag*sin(2*pi/(i*2)));
		for (int j=0;j<n;j+=(i<<1))
		{
			com w(1,0);
			for (int k=0;k<i;k++,w*=wn)
			{
				com x=a[j+k],y=w*a[j+k+i];
				a[j+k]=x+y;
				a[j+k+i]=x-y;
			}
		}
	}
	if (flag==-1) for (int i=0;i<n;i++) a[i]/=n;
}

void init()
{
	char str[MAXN];
	scanf("%d",&n);
	scanf("%s",str);
	for (int i=0;i<n;i++) a[i]=str[n-1-i]-'0';
	scanf("%s",str);
	for (int i=0;i<n;i++) b[i]=str[n-1-i]-'0';
}
```

```
void solve()
{
	m=n<<1;//相乘后的位数是原来的 2 倍
	get_bit();
	get_Rtable();//求逆序表：末位为 0，直接为其前一半逆序表的值右移一位，末位为 1，
在最高位添加 1
	FFT(a,1),FFT(b,1);//分别将 a 与 b 的系数表达式转为点值表达式
	multi(a,b);//点值表达式相乘
	FFT(a,-1);//将相乘后的点值表达式转为系数表达式

}

void print()
{
	for(int i=0;i<m;i++) c[i]=(int)(a[i].real()+0.5);
	for (;c[m-1]==0;m--); //把前置的 0 清空
	for (int i=0;i<m;i++)
	{
		if (c[i]>=10)
		{
			c[i+1]+=c[i]/10;
			c[i]%=10;
			if (i==m-1) m++;
		}
	}
	for (int i=m-1;i>=0;i--) printf("%d",c[i]);
}

int main()
{
	init();
	solve();
	print();
	return 0;
}
```

3. FWT

```
#include <bits/stdc++.h>
using namespace std;

const int N = 2048 + 5;

typedef long long ll;
typedef unsigned long long llu;
```

```
void FWT(ll a[], int n) {
    for(int d = 1; d < n; d <<= 1) {
        for(int m = d << 1, i = 0; i < n; i += m) {
            for(int j = 0; j < d; ++ j) {
                ll x = a[i + j], y = a[i + j + d];
                a[i + j] = x + y;
                a[i + j + d] = x - y;
            }
        }
    }
}

void UFWT(ll a[], int n) {
    for(int d = 1; d < n; d <<= 1) {
        for(int m = d << 1, i = 0; i < n; i += m) {
            for(int j = 0; j < d; ++ j) {
                ll x = a[i + j], y = a[i + j + d];
                a[i + j] = (x + y) / 2;
                a[i + j + d] = (x - y) / 2;
            }
        }
    }
}

ll a[3][N], cnt[3][N];
ll pre[N];

void update(int n, int m, int d, int t) {
    for(int i = 0; i <= n && i + d <= m; ++ i)
        cnt[t][i ^ (i + d)] ++;
    if(d) {
        for(int i = 0; i <= m && i + d <= n; ++ i)
            cnt[t][i ^ (i + d)] ++;
    }
}

int main() {
        int T, kase = 1;
    int ib, ab, gb, ig, ag, gg;
    scanf("%d", &T);
    while(T --) {
        memset(pre, 0, sizeof(pre));
        memset(a, 0, sizeof(a));
```

```
        memset(cnt, 0, sizeof(cnt));
        scanf("%d%d%d%d%d%d", &ib, &ab, &gb, &ig, &ag, &gg);
        int maxs = max({ib, ab, gb, ig, ag, gg});
        int tmp = maxs;
        int bl = 1;
        llu ans = 0;
        while(tmp) {
            tmp >>= 1; bl <<= 1;
        }
        for(int d = 0; d <= maxs; ++ d) {
            update(ib, ig, d, 0);
            update(ab, ag, d, 1);
            update(gb, gg, d, 2);
            for(int i = 0; i < bl; ++ i) {
                a[0][i] = cnt[0][i];
                a[1][i] = cnt[1][i];
                a[2][i] = cnt[2][i];
            }
            FWT(a[0], bl);
            FWT(a[1], bl);
            FWT(a[2], bl);
            for(int i = 0; i < bl; ++ i) {
                a[0][i] = a[0][i] * a[1][i] * a[2][i];
            }
            UFWT(a[0], bl);
            for(int i = 0; i < bl; ++ i) {
                ll k = 1ll * (d ^ i);
                ans += 1ull * (a[0][i] - pre[i]) * k;
                pre[i] = a[0][i];
            }
        }
        cout << "Case #" << kase ++ << ": " << ans << endl;
    }
    return 0;
}
```

4. NTT

```
#include <bits/stdc++.h>
using namespace std;

const int N = 2e5 + 5;
const int MOD = 998244353;
```

```
int fpow(int a,int b) {
    int rtn=1;
    while(b) {
        if(b&1)rtn=1ll*rtn*a%MOD;
        a=1ll*a*a%MOD;b>>=1;
    }
    return rtn;
}

inline int rfadd(int a,int b) {
    a+=b;if(a>=MOD)a-=MOD;
    return a;
}
inline void fadd(int&a,int b) {
    a+=b;if(a>=MOD)a-=MOD;
}
inline int rfsub(int a,int b) {
    a-=b;if(a<0)a+=MOD;
    return a;
}

int a[8*N],b[8*N],snw[8*N],fnw[8*N];
struct NTT {
    int siz;
    void init(int n) {
        siz=1;
        for (;siz<(n<<1);siz<<=1);
        for (int i=0;i<siz;++i) a[i]=b[i]=0;
    }
    void ntt(int *p,int f) {
        for (int i=0,j=0;i<siz;++i) {
            if (i<j) swap(p[i],p[j]);
            for (int k=siz>>1;(j^=k)<k;k>>=1);
        }
        for (int i=2;i<=siz;i<<=1) {
            int nw=snw[i];
            if (f==-1) nw=fnw[i];
            for (int j=0,m=i>>1;j<siz;j+=i) {
                for (int k=0,w=1;k<m;++k) {
                    int t=1ll*p[j+k+m]*w%MOD;
                    p[j+k+m]=rfsub(p[j+k],t);
                    p[j+k]=rfadd(p[j+k],t);
                    w=1ll*w*nw%MOD;
                }
```

```
            }
        }
        if (f==-1) {
            int inv=fpow(siz,MOD-2);
            for (int i=0;i<siz;++i) p[i]=1ll*p[i]*inv%MOD;
        }
    }
    void fmul() {
        ntt(a,1);ntt(b,1);
        for (int i=0;i<siz;++i) a[i]=1ll*a[i]*b[i]%MOD;
        ntt(a,-1);
    }
};

int inv[4*N],fac[4*N];
void init() {
    for (int i=2;i<8*N;++i) {
        if (i!=2&&(MOD-1)/i==(MOD-1)/(i-1)) {
            snw[i]=snw[i-1]; fnw[i]=fnw[i-1];
        } else {
            snw[i]=fpow(3,(MOD-1)/i);
            fnw[i]=fpow(snw[i],MOD-2);
        }
    }
    inv[0]=inv[1]=1;
    fac[0]=fac[1]=1;
    for (int i=2;i<4*N;++i) {
        fac[i]=1ll*fac[i-1]*i%MOD;
        inv[i]=1ll*(MOD-MOD/i)*inv[MOD%i]%MOD;
    }
    for (int i=2;i<4*N;++i) inv[i]=1ll*inv[i-1]*inv[i]%MOD;
}

int C(int n,int m) {
    return 1ll*fac[n]*inv[m]%MOD*inv[n-m]%MOD;
}

/*****************************/
//#define yswness
#ifdef yswness
void dg() {
    cout << endl;
}
template<typename T, typename... A>
```

```
void dg(T a, A... x) {
    cout << a << " ";
    dg(x...);
}
#endif

/*****************************/

struct rect {
    int xl,xr,yl,yr;
} p[N];

int na[N],nb[N],h[N],ta[N],tb[N];
unordered_map<int,int> f[N];

void getrect(int l,int r) {
    if (l>r) return;
    int mid=(l+r)>>1;
    p[mid]=(rect){mid,r,h[l-1]+1,h[mid]};
    getrect(l,mid-1);
    getrect(mid+1,r);
}

int F(int n,int m) {
    int t=0;
    if (m>1) t=f[n][m-1];
    fadd(t,f[n-1][m]);
    return f[n][m]=t;
}

void cal(int k) {
    int xl=p[k].xl,yl=p[k].yl;
    int xr=p[k].xr,yr=p[k].yr;
    if (yl>yr) return;
    if (xl==1) {
        for (int i=xl;i<=xr;++i) {
            for (int j=yl;j<=yr;++j) f[i][j]=C(i+j-2,i-1);
        } return;
    }
    int nx=xr-xl+1,ny=yr-yl+1;
    for (int i=yl;i<=yr;++i) na[i-yl]=f[xl-1][i];//,dg(xl-1,i,na[i-yl]);
    for (int i=xl;i<=xr;++i) nb[i-xl]=f[i][yl-1];//,dg(i,yl-1,nb[i-xl]);
    for (int i=0;i<nx;++i) ta[i]=0;
    for (int i=0;i<ny;++i) tb[i]=0;
```

```
    //dg(nx,ny);
    NTT qls;
    // first case:left to top
    qls.init(nx+ny-1);
    for (int i=0;i<ny;++i) a[i]=1ll*na[i]*inv[yr-yl-i]%MOD;
    for (int i=0;i<nx+ny-1;++i) b[i]=fac[i];
    qls.fmul();
    for (int i=ny-1;i<nx+ny-1;++i) fadd(ta[i-ny+1],1ll*a[i]*inv[i-ny+1]%MOD);
    //for (int i=0;i<nx;++i) dg(xl+i,yr,ta[i]);
    // second case:left to right
    qls.init(ny);
    for (int i=0;i<ny;++i) a[i]=na[i];
    for (int i=0;i<ny;++i) b[i]=C(xr-xl+i,xr-xl);
    qls.fmul();
    for (int i=0;i<ny;++i) fadd(tb[i],a[i]);
    // third case:bottom to top
    qls.init(nx);
    for (int i=0;i<nx;++i) a[i]=nb[i];
    for (int i=0;i<nx;++i) b[i]=C(yr-yl+i,yr-yl);
    qls.fmul();
    for (int i=0;i<nx;++i) fadd(ta[i],a[i]);
    // fourth case:bottom to right
    qls.init(nx+ny-1);
    for (int i=0;i<nx;++i) a[i]=1ll*nb[i]*inv[xr-xl-i]%MOD;
    for (int i=0;i<nx+ny-1;++i) b[i]=fac[i];
    qls.fmul();
    for (int i=nx-1;i<nx+ny-1;++i) fadd(tb[i-nx+1],1ll*a[i]*inv[i-nx+1]%MOD);
    // finished
    for (int i=0;i<nx;++i) f[xl+i][yr]=ta[i];//,dg(xl+i,yr,ta[i]);
    for (int i=0;i<ny;++i) f[xr][yl+i]=tb[i];//,dg(xr,yl+i,tb[i]);
}

void solve() {
    int n;scanf("%d",&n);
    for (int i=1;i<=n;++i) scanf("%d",h+i);
    h[0]=0; getrect(1,n);
    for (int i=1;i<=n;++i) f[i].clear();
    for (int i=1;i<=n;++i) cal(i);
    int ans=0;
    for (int i=1;i<=h[n];++i) fadd(ans,f[n][i]);
    printf("%d\n",ans);
}

int main() {
```

```
/**Constant optimized version; 19/27/8**/
#ifdef yswness
    freopen("in", "r", stdin);
    //freopen("out", "w", stdout);
#endif
    init();
    int T;scanf("%d",&T);
    for (;T;--T) solve();
    return 0;
}
```

5. Lucas

$C(n,m) = C(n/p,m/p) * C(n\%p,m\%p) \% p;$

6. 拉格朗日

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

using namespace std;

typedef long long ll;

const ll MOD = 1e9 + 7;

int remod(ll x) {
    x %= MOD;
    if(x < 0) x += MOD;
    return x;
}

void fadd(int& a, int b) {
    a += b;
    if(a >= MOD) a -= MOD;
}

//连续 x，x 起始点为 x0
struct PolyInter {
    ll x0; int deg;
    vector <int> buf, inv, val;

    void ini(const vector <int>& v, ll in = 0)
    {
        deg = v.size(); buf = val = v; x0 = in;
        inv.resize(max(deg, 2));
        inv[1] = 1;
```

```
        for(int i = 2; i < deg; ++ i)
            inv[i] = 1ll * (MOD - MOD/i) * inv[MOD%i] % MOD;
    }

    int eval(ll x)
    {
        ll b = 1;
        for(int i = 1; i < deg; ++ i) {
            b = b * remod(x - x0 - i + 1) % MOD * inv[i] % MOD;
            buf[i] = val[i] * b % MOD;
        }
        b = 1;
        int res = buf[deg - 1];
        for(int i = deg - 2; i >= 0; -- i) {
            b = (MOD - b) * remod(x - x0 - i - 1) % MOD * inv[deg - i - 1] % MOD;
            res += b * buf[i] % MOD;
            if(res >= MOD) res -= MOD;
        }
        return res;
    }
};

ll fpow(ll a, int b) {
    ll rtn = 1;
    while(b) {
        if(b & 1) rtn = (rtn * a) % MOD;
        a = (a * a) % MOD;
        b >>= 1;
    }
    return    rtn;
}

vector <int> a, b;

int main() {
    int n;
    scanf("%d", &n);
    b.resize(n + 1);
    for (int i = 0; i <= n; ++ i) {
        scanf("%d", &b[i]);
    }
    PolyInter p;
    p.ini(b);
    int x;
```

```
    scanf("%d", &x);
    printf("%d\n", p.eval(x));
    return 0;
}
```

7. 莫比乌斯

```
#include <bits/stdc++.h>
using namespace std;

const int N = 1e6 + 5;

bool vis[N];
int prime[N];
int phi[N];
int mu[N];
int tot = 0;

void init()
{
    mu[1] = phi[1] = 1;
    for (int i = 2; i < N; ++ i) {
        if (!vis[i]) {
            prime[tot ++] = i;
            phi[i] = i - 1;
            mu[i] = -1;
        }
        for (int j = 0; j < tot && i * prime[j] < N; ++ j) {
            vis[i * prime[j]] = 1;
            if (i % prime[j] == 0) {
                mu[i * prime[j]] = 0;
                phi[i * prime[j]] = phi[i] * prime[j];
                break;
            } else {
                mu[i * prime[j]] = -mu[i];
                phi[i * prime[j]] = phi[i] * (prime[j] - 1);
            }
        }
    }
}

int main()
{
    init();
}
```

狄利克雷卷积 $(f * g)(n) = \sum_{d|n} f(d) g(\frac{n}{d})$

常见的积性函数:
欧拉函数
莫比乌斯函数
单位函数
不变函数
幂函数
因子个数函数
因子和函数
因子函数
狄利克雷卷积单位元

常用公式:

1. 莫比乌斯函数与不变函数的狄利克雷卷积为狄利克雷卷积单位元

$$\mu * 1\ (n) = [n == 1]$$

2. 单位函数与莫比乌斯函数的狄利克雷卷积为欧拉函数（精髓）

$$Id * \mu\ (n) = \varphi(n)$$

3. 欧拉函数与不变函数的狄利克雷卷积为单位函数

$$\varphi * 1\ (n) = id(n)$$

若存在函数$h(n) = f * g(n)$，且 f、g 均为可积函数，则有

$$g(1)S(n) = \sum h(i) - \sum g(i) * S(n/i)$$

8. 欧拉定理

一个联通平面图 G 由 v 个顶点、e 条边、f 个面，那么有

v-e+f=2

a^phi(m)%m = 1

a^(b%phi(c)+phi(c))%c = a^b%c

9. 约瑟夫环

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

int main()
{
   int T, kase = 1;
   scanf("%d", &T);
   while(T --) {
      ll n, m, k;
      scanf("%I64d%I64d%I64d", &n, &m, &k);
      printf("Case #%d: ", kase ++);
      if(m < k) {
```

```
            ll f1 = (k-1)%(n-m+1);
            for(ll i = n-m+2; i <= n; ++ i) {
                f1 = (f1 + k) % i;
            }
            printf("%I64d\n", f1 + 1);
        } else {
            if(k == 1) {
                printf("%I64d\n", m);
                continue;
            }
            ll f1 = (k-1)%(n-m+1);
            for(ll i = n-m+2, j = i; i <= n; i = j+1) {
                ll sp = (i-1-f1)/(k-1);
                if((i-1-f1)%(k-1)!=0) sp++;
                if(i+sp-1>=n) {
                    f1=(f1+(n-i+1)*k)%n; break;
                }
                f1 = (f1+sp*k)%(i+sp-1);
                j = i+sp-1;
            }
            printf("%I64d\n", f1 + 1);
        }
    }
    return 0;
}
```

f(N, M) = (f(N-1, M) + M) % N;

10. Polya 计数

L=[m^(c(p1)) + m^(c(p2)) + … + m^(c(pn))]/|G|

11. 线性递推

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

const int P = 1e9 + 7;

int linear_recurrence(ll n, int m, vector<int> &a, vector<int> &c) {
    if (n < m) return (a[n] + P) % P;
    vector<ll> v(m, 0), u(m<<1, 0);
    v[0] = 1;
    for (ll x = 0, W = n ? 1ll<<(63 - __builtin_clzll(n)) : 0; W; W >>= 1, x <<= 1) {
        fill(u.begin(), u.end(), 0);
        int b = !!(n & W); if (b) x ++;
        if (x < m) u[x] = 1;
```

```
        else {
            for (int i = 0; i < m; ++ i) {
                for (int j = 0; j < m; ++ j) {
                    (u[i + b + j] += v[i] * v[j]) %= P;
                }
            }
            for (int i = 2*m - 1; i >= m; -- i) {
                for (int j = 0; j < m; ++ j) {
                    (u[i - m + j] += c[j] * u[i]) %= P;
                }
            }
        }
        copy(u.begin(), u.begin() + m, v.begin());
    }
    ll ans = 0;
    for (int i = 0; i < m; ++ i)
        (ans += v[i] * a[i]) %= P;
    return (ans + P) % P;
}

ll fpow(ll a, ll b) {
    ll rtn = 1;
    while (b) {
        if (b & 1) rtn = rtn * a % P;
        a = a * a % P; b >>= 1;
    }
    return rtn;
}

int T; ll k, n;
vector<int> a, c;

int main() {
    scanf("%d", &T);
    while (T --) {
        scanf("%lld%lld", &k, &n);
        if (n == -1) printf("%lld\n", fpow(k + 1, P - 2) * 2 % P);
        else {
            a.clear(); c.clear();
            a.resize(k, 0); c.resize(k, 0);
            a[0] = 1; ll kk = fpow(k, P - 2);
            for (int i = 1; i < k; ++ i) {
                for (int j = 0; j < i; ++ j) a[i] = 1ll * (a[i] + a[j]) % P;
                a[i] = 1ll * a[i] * kk % P;
```

```
            }
            for (int i = 0; i < k; ++ i) c[i] = kk;
            int ans = linear_recurrence(n, k, a, c);
            printf("%d\n", ans);
        }
    }
    return 0;
}
```

12. 线性基插入查找

```
#include <bits/stdc++.h>
using namespace std;

const int N = 1e6 + 5;

int ind[N][31], f[N][31];

void Insert(int x, int p)
{
    int np = p;
    for (int i = 30; i >= 0; -- i) {
        f[p][i] = f[p - 1][i];
        ind[p][i] = ind[p - 1][i];
    }
    for (int i = 30; i >= 0; -- i) {
        if ((x >> i) & 1) {
            if (!ind[p][i]) {
                ind[p][i] = np; f[p][i] = x;
                break;
            }
            if (ind[p][i] < np) {
                swap(ind[p][i], np);
                swap(f[p][i], x);
            }
            x ^= f[p][i];
        }
    }
}

int main()
{
    int T;
    scanf("%d", &T);
    while (T --) {
```

```
        int n, m;
        scanf("%d%d", &n, &m);
        for (int i = 1; i <= n; ++ i) {
            int x;
            scanf("%d", &x);
            Insert(x, i);
        }
        int ans = 0;
        for (int i = 0; i < m; ++ i) {
            int cmd;
            scanf("%d", &cmd);
            if (cmd == 0) {
                int l, r;
                scanf("%d%d", &l, &r);
                l ^= ans; l = l % n + 1;
                r ^= ans; r = r % n + 1;
                if (l > r) swap(l, r);
                ans = 0;
                for (int j = 30; j >= 0; -- j) {
                    if (ind[r][j] >= l && (ans ^ f[r][j]) > ans) {
                        ans ^= f[r][j];
                    }
                }
                printf("%d\n", ans);
            } else {
                int x;
                scanf("%d", &x);
                x ^= ans;
                Insert(x, n + 1);
                n ++;
            }
        }
    }
    return 0;
}
```

13. 求线性基的交

```
LinearBasis Merge(LinearBasis A,LinearBasis B) {
        LinearBasis All , C , D;
        All.clear();
        C.clear();
        D.clear();
        for (int i = 60;i >= 0;i--) All.basis[i] = A.basis[i];
```

```
        for (int i = 60; i >= 0; i--) {
            if (B.basis[i]) {
                ll v = B.basis[i] , k = 1ll << i;
                bool can = true;
                for (int j = 60; j >= 0; j--) {
                    if (v & (1ll << j)) {
                        if (All.basis[j]) {
                            v ^= All.basis[j];
                            k ^= D.basis[j];
                        } else {
                            can = false;
                            All.basis[j] = v;
                            D.basis[j] = k;
                            break;
                        }
                    }
                }

                if (can) {
                    ll v = 0;
                    for (int j = 60; j >= 0; j--) {
                        if (k & (1ll << j)) {
                            v ^= B.basis[j];
                        }
                    }
                    C.insert(v);
                }
            }
        }
        C.build();
        return C;
}
```

14. 辗转相除法求最佳分数逼近

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

// get the best fraction which is large than a1/b1 and less than a2/b2
// x is smallest

void exp_gcdll(ll a1,ll b1,ll a2,ll b2,ll& x,ll& y) {
    ll t = a1/b1+1;
    ll q = (a2+b2-1)/b2-1;
```

```
    if (t <= q) {
        x = t; y = 1; return ;
    }
    a1 -= (t-1)*b1;
    a2 -= (t-1)*b2;
    exp_gcdll(b2,a2,b1,a1,y,x);
    x += (t-1)*y;
}

int main() {
    ll a, b, c, d, x, y;
    cin >> a >> b >> c >> d;
    exp_gcdll(a, b, c, d, x, y);
    cout << x << "/" << y << endl;
    return 0;
}
```