序列中两两之差

CF1398G

把题目化简之后，就是给一个序列 a_i 要能高效地得到 $a_i - a_j$ 构成的集合。

构造两个生成函数 $\sum x^{a_i}$ 和 $\sum x^{-a_i}$那么这两个多项式相乘得到的答案 $f(x)$ 中如果 x^i 的系数不为 0 则 i 可以被表示为序列中某两个数的差。

```
/*
#pragma GCC optimize(2)
#pragma GCC optimize(3,"Ofast","inline")
*/
#include<bits/stdc++.h>
#define ALL(x) (x).begin(),(x).end()
#define ll long long
#define db double
#define ull unsigned long long
#define pii_ pair<int,int>
#define mp_ make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<"; "<<#b<<" = "<<b<<endl
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
const int maxn = 1e6+5;
const int M = 998244353;
inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}
ll qpow(ll a,ll b) {ll s=1;while(b){if(b&1)s=s*a%M;a=a*a%M;b>>=1;}return
s;}

int rev[maxn],f[maxn];
ll A[maxn],B[maxn];
void trans(ll a[],int lim,int type)
{
    rep(i,1,lim-1) if(i<rev[i]) swap(a[i],a[rev[i]]);
    for(int mid=1;mid<lim;mid<<=1){
        ll wn = qpow(3,(M-1)/mid/2);
        if(type==-1) wn = qpow(wn,M-2);
        for(int R=mid<<1,j=0;j<lim;j+=R){
            ll w = 1;
            for(int k=0;k<mid;k++,w=w*wn%M){
                ll x=a[j+k],y=w*a[j+mid+k]%M;
                a[j+k] = (x+y)%M;
                a[j+mid+k] = (x-y+M)%M;
```

```
            }
        }
    }
    if(type==-1){
        ll inv = qpow(lim,M-2);
        rep(i,0,lim) a[i] = a[i]*inv%M;
    }
}

int main()
{
    fastio();
    memset(f,-1,sizeof(f));
    int n,x,y;
    cin>>n>>x>>y;
    rep(i,0,n){
        int a;cin>>a;
        A[a] = 1;
        B[x-a] = 1;
    }
    int lim=1,l=0;
    while(lim<=(int)4e5) lim<<=1,l++;
    rep(i,1,lim-1) rev[i] = (rev[i>>1]>>1) | ((i&1)<<(l-1));
    trans(A,lim,1);trans(B,lim,1);
    rep(i,0,lim) A[i] = A[i] * B[i] %M;
    trans(A,lim,-1);
    rep(i,x+1,lim){
        if(2*(y + i-x) > (int)1e6) break;
        if(A[i] > 0) f[2*(y + i-x)] = 2*(y + i-x);
    }
    rep(i,4,1000000){ if(f[i]==-1) continue;
        for(int j = i+i;j<=1000000;j+=i){
            f[j] = max(f[j],f[i]);
        }
    }
    int q; cin>>q;
    while(q--){
        int l; cin>>l;
        cout<<f[l]<<" ";
    }
    return 0;
}
```

字符串匹配

先考虑如何用 FFT 做和 KMP 一样的事：

记 s 串为长度为 m 的模式串，t 串为长度为 n 的文本串，下标均从 0 开始。目标是找出所有 x 满足 $\forall i\in [0,m),t_{x-m+i+1} = s_i$。

上述条件可用一个式子来表示 $\sum _{i=0}^{m-1} (s_i - t_{x-m+i+1})^2 = 0$，展开后为 $\sum _{i=0}^{m-1} s_i^2 + t_{x-m+i+1}^2 - 2\cdot s_i\cdot t_{x-m+i+1}$。

两个平方项都可以通过前缀和得到。乘积项转换一下就会变成一个卷积的形式：把 s 串翻转一下得到 rs 串，于是乘积项就是 $\sum _{i=0}^{m-1} 2\cdot rs_{m-i-1} \cdot t_{x-m+i+1} = \sum _{i=0}^{m-1} 2\cdot rs_i \cdot t_{x-i}$，所以这里就可以用$\text{FFT}$处理的到。

最后的判断条件：卷积之后的多项式为 $f(x)$，在 t 串的 $x0$ 位置匹配当且仅当 $f(x) + pres[m-1] + pret[x] - pret[x-m-2] = 0$，总复杂度为 $O(nlogn)$。

虽然 FFT 多了一个 log 的复杂度，但有些匹配是 KMP 无法做但是 FFT 可以做。

P4173

在正常匹配的基础上扩大了字符集的范围，多了一种 $*$ 字符（可以匹配任何字符）。这样之后就不能用 KMP 了，因为 KMP 需要满足一种等价关系，而通配符 $*$ 的存在就不满足等价关系，$a = * 且 b = *$ 但没有传递性 $a = b$。

考虑用上述一样的方法来构造多项式函数 $\sum_{i=0}^{m-1} (s_i - t_{x-m+i+1})^2 \cdot s_i \cdot t_i $，这样就可以得到答案了。

```
#pragma GCC optimize(2)
#pragma GCC optimize(3,"Ofast","inline")

#include<bits/stdc++.h>
#define ALL(x) (x).begin(),(x).end()
#define ll long long
#define db double
#define ull unsigned long long
#define pii_ pair<int,int>
#define mp_ make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<"; "<<#b<<" = "<<b<<endl
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
const int maxn = 12e5+5;
const int M = 998244353;
inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}
ll qpow(ll a,ll b) {ll s=1;while(b){if(b&1)s=s*a%M;a=a*a%M;b>>=1;}return
s;}
int rev[maxn];
```

```
ll A[maxn],B[maxn],C[maxn];
void trans(ll a[],int lim,int type)
{
    rep(i,1,lim-1) if(i<rev[i]) swap(a[i],a[rev[i]]);
    for(int mid=1;mid<lim;mid<<=1){
        ll wn = qpow(3,(M-1)/mid/2);
        if(type==-1) wn = qpow(wn,M-2);
        for(int R=mid<<1,j=0;j<lim;j+=R){
            ll w = 1;
            for(int k=0;k<mid;k++,w=w*wn%M){
                ll x=a[j+k],y=w*a[j+mid+k]%M;
                a[j+k] = (x+y)%M;
                a[j+mid+k] = (x-y+M)%M;
            }
        }
    }
    if(type==-1){
        ll inv = qpow(lim,M-2);
        rep(i,0,lim) a[i] = a[i]*inv%M;
    }
}
char t[maxn],s[maxn];
int idt[maxn],ids[maxn];
int main()
{
    fastio();
    int n,m; cin>>n>>m;
    cin>>t>>s;
    reverse(t,t+n);
    rep(i,0,n-1) idt[i] = t[i]=='*'?0:t[i]-'a'+1;
    rep(i,0,m-1) ids[i] = s[i]=='*'?0:s[i]-'a'+1;

    int lim = 1,l = 0;
    while(lim<=n+m) lim<<=1,l++;
    rep(i,1,lim-1) rev[i] = (rev[i>>1]>>1) | ((i&1)<<(l-1));

    rep(i,0,n-1){
        A[i] = idt[i];
    }
    rep(i,0,m-1){
        B[i] = ids[i]*ids[i]*ids[i];
    }
    trans(A,lim,1);trans(B,lim,1);
    rep(i,0,lim) C[i] = C[i] + A[i] * B[i];

    memset(A,0,sizeof(A));
    memset(B,0,sizeof(B));
    rep(i,0,n-1){
        A[i] = idt[i]*idt[i];
    }
```

```
    rep(i,0,m-1){
        B[i] = ids[i]*ids[i];
    }
    trans(A,lim,1);trans(B,lim,1);
    rep(i,0,lim) C[i] = C[i] - 2LL * A[i] * B[i];

    memset(A,0,sizeof(A));
    memset(B,0,sizeof(B));
    rep(i,0,n-1){
        A[i] = idt[i]*idt[i]*idt[i];
    }
    rep(i,0,m-1){
        B[i] = ids[i];
    }
    trans(A,lim,1);trans(B,lim,1);
    rep(i,0,lim) C[i] = C[i] + A[i] * B[i];

    trans(C,lim,-1);

    vector<int> ans;
    rep(i,n-1,m-1){
        if(C[i] == 0) ans.pb(i-n+2);
    }
    cout<<ans.size()<<endl;
    for(int x:ans) cout<<x<<" ";
    return 0;
}
```

CF1334G

在普通匹配的基础上添加条件 $p(s_i) = t_i$ 时也算匹配，p 是题目给的一个置换。

同样不满足传递性，因此只能用 FFT 匹配。

```
#pragma GCC optimize(2)
#pragma GCC optimize(3,"Ofast","inline")
#include<bits/stdc++.h>
#define ALL(x) (x).begin(),(x).end()
#define ll long long
#define db double
#define ull unsigned long long
#define pii_ pair<int,int>
#define mp_ make_pair
#define pb push_back
#define fi first
#define se second
#define rep(i,a,b) for(int i=(a);i<=(b);i++)
#define per(i,a,b) for(int i=(a);i>=(b);i--)
#define show1(a) cout<<#a<<" = "<<a<<endl
#define show2(a,b) cout<<#a<<" = "<<a<<"; "<<#b<<" = "<<b<<endl
```

```
using namespace std;
const ll INF = 1LL<<60;
const int inf = 1<<30;
const int maxn = 1e6+5;
const int M = 998244353;
inline void fastio() {ios::sync_with_stdio(false);cin.tie(0);cout.tie(0);}
ll qpow(ll a,ll b) {ll s=1;while(b){if(b&1)s=s*a%M;a=a*a%M;b>>=1;}return
s;}

int rev[maxn];
ll A[maxn],B[maxn],C[maxn],T4[maxn],val[30],p[30];
char s[maxn],t[maxn];
void trans(ll a[],int lim,int type)
{
    rep(i,1,lim-1) if(i<rev[i]) swap(a[i],a[rev[i]]);
    for(int mid=1;mid<lim;mid<<=1){
        ll wn = qpow(3,(M-1)/mid/2);
        if(type==-1) wn = qpow(wn,M-2);
        for(int R=mid<<1,j=0;j<lim;j+=R){
            ll w = 1;
            for(int k=0;k<mid;k++,w=w*wn%M){
                ll x=a[j+k],y=w*a[j+mid+k]%M;
                a[j+k] = (x+y)%M;
                a[j+mid+k] = (x-y+M)%M;
            }
        }
    }
    if(type==-1){
        ll inv = qpow(lim,M-2);
        rep(i,0,lim) a[i] = a[i]*inv%M;
    }
}
void add(ll &x,ll y)
{
    x += y;
    if(x>=M) x-=M;
}
bool check(int k)
{
    rep(i,0,k) rep(j,i+1,k) if(val[i]==val[j]) return 0;
    return 1;
}
int main()
{
    fastio();srand(time(NULL));
    rep(i,0,25){ //val[i] = i;
        val[i] = rand()%M;
        while(!check(i)) add(val[i],1);
    }
    rep(i,0,25) cin>>p[i],p[i]=val[p[i]-1];
```

```
    cin>>s>>t;
    int m = strlen(s),n = strlen(t);
    rep(i,0,m-1) s[i] -= 'a';
    rep(i,0,n-1) t[i] -= 'a';
    reverse(s,s+m);

    int lim = 1,l = 0;
    while(lim <= n+m) lim<<=1,l++;
    rep(i,1,lim-1) rev[i] = (rev[i>>1]>>1) | ((i&1)<<(l-1));

    rep(i,0,m-1) A[i] = (-2LL*val[s[i]]*val[s[i]]%M*p[s[i]]%M -
2LL*val[s[i]]*p[s[i]]%M*p[s[i]]%M)%M;
    rep(i,0,n-1) B[i] = val[t[i]];
    trans(A,lim,1);trans(B,lim,1);
    rep(i,0,lim) add(C[i],A[i]*B[i]%M);

    memset(A,0,sizeof(A));memset(B,0,sizeof(B));
    rep(i,0,m-1) A[i] = (1LL*val[s[i]]*val[s[i]]%M +
4LL*val[s[i]]*p[s[i]]%M + 1LL*p[s[i]]*p[s[i]]%M)%M;
    rep(i,0,n-1) B[i] = val[t[i]] * val[t[i]] % M;
    trans(A,lim,1);trans(B,lim,1);
    rep(i,0,lim) add(C[i],A[i]*B[i]%M);

    memset(A,0,sizeof(A));memset(B,0,sizeof(B));
    rep(i,0,m-1) A[i] = (-2LL*val[s[i]] -2LL*p[s[i]])%M;
    rep(i,0,n-1) B[i] = val[t[i]] * val[t[i]] %M * val[t[i]] %M;
    trans(A,lim,1);trans(B,lim,1);
    rep(i,0,lim) add(C[i],A[i]*B[i]%M);

    trans(C,lim,-1);

    rep(i,0,n-1){
        if(i==0) T4[i] = val[t[i]]*val[t[i]]%M*val[t[i]]%M*val[t[i]]%M;
        else T4[i] = (T4[i-1] +
val[t[i]]*val[t[i]]%M*val[t[i]]%M*val[t[i]]%M)%M;
    }

    ll sps = 0;
    rep(i,0,m-1){
        add(sps,val[s[i]] * val[s[i]] %M *p[s[i]]%M *p[s[i]]%M);
    }

    rep(i,m-1,n-1){
        ll res = (T4[i] - (i==m-1?0:T4[i-m]) + C[i] + sps)%M; //show1(res);
        if(res==0) cout<<1;
        else cout<<0;
    }

    return 0;
}
```