```
1.segmented_seive
bitset<mx7>visited;
int prime[mx7], sz=0,cs=0;
vector<ll>sg;
void seive()
{
    for(ll i=4; i<mx7; i+=2)visited[i]=1;
    for(ll i=3; i*i<mx7; i+=2)
    {
        if(visited[i])continue;
        for(ll j=i*i; j<mx7; j+=(2*i))
        {
            visited[j]=1;
        }
    }
    prime[0]=2;
    int cnt=1;
    for(int i=3; i<mx7; i+=2)
    {
        if(!visited[i])prime[cnt++]=i;
    }
    sz=cnt;
}
void segmented_seive(ll l,ll r)
{
    sg.clear();
    ll root=sqrt(r)+1;

    for(ll i=l; i<=r; i++)sg.pb(i);
    if(l==0)sg[1]=0;
    else if(l==1)sg[0]=0;
```



```

    for(ll i=0;  prime[i]<=root; i++)
    {
        ll d=prime[i];
        ll start=d*d;
        if(start<l)start=((l+d-1)/d)*d;

        for(ll j=start; j<=r; j+=d)
        {
            sg[j-l]=0;
        }
    }
}
vector<ll>v;
void input()
{
    ll n,m,cnt=0,d,l,r;
    sc2(l,r);
    segmented_seive(l,r);
    for(ll i=l; i<=r; i++)
    {
        if(sg[i-l]!=0)v.pb(sg[i-l]);
    }
    //for(int i=0;i<v.size();i++)cout<<v[i]<<" ";
    ll lenth=v.size();
    printf("Case %d: %lld\n",++cs,lenth);
    v.clear();
}
2.segment_tree_with_Ordered_set (L to R kth max)
#include<bits/stdc++.h>
#include<ext/pb_ds/assoc_container.hpp>
```



```
using namespace std;
#include<ext/pb_ds/tree_policy.hpp>
using namespace __gnu_pbds;
typedef tree<pair<ll,ll>,null_type,less< pair<ll,ll>
>,rb_tree_tag,
        tree_order_statistics_node_update>ordered_set;
ordered_set tr[4*mx5];
ll a[mx6];
void build(int u,int b,int e)
{
    if(b==e)
    {
        tr[u].insert({a[b],b});
        return;
    }
    ll mid=(b+e)/2;
    build(2*u,b,mid);
    build(2*u+1,mid+1,e);
    for(int i=b; i<=e; i++)tr[u].insert({a[i],i});
}
ll quary(int u,int b,int e,int i,int j,int val,int id)
{
    if(e<i or b>j)return 0;
    else if(b>=i and e<=j)
    {
        ll sz=tr[u].order_of_key({val+1,id});
        return sz;
    }
    int   mid=(b+e)/2;
    ll left=quary(2*u,b,mid,i,j,val,id);
    ll right=quary((2*u)+1,mid+1,e,i,j,val,id);
```



```
    return  left+right;
}
void  update_delete (int u,int b,int e,int i,int j,int
val,int id)
{
    if(e<i or b>j)return;
    else if(b>=i and e<=j)
    {
        tr[u].erase(tr[u].find({val,id}));
        return;
    }
    int mid=(b+e)/2;
    update_delete(2*u,b,mid,i,j,val,id);
    update_delete((2*u)+1,mid+1,e,i,j,val,id);
    tr[u].erase(tr[u].find({val,id}));
}
void  update_add(int u,int b,int e,int i,int j,int x,int
id)
{
    if(e<i or b>j)return;
    else if(b>=i and e<=j)
    {
        tr[u].insert({x,id});
        return;
    }
    int mid=(b+e)/2;
    update_add(2*u,b,mid,i,j,x,id);
    update_add((2*u)+1,mid+1,e,i,j,x,id);
    tr[u].insert({x,id});
}

```

```
void input()
{
    ll n,m,res=0,l,r;
    sc(n);
    for(int i=1; i<=n; i++)
    {
       sc(a[i]);
       update_add(1,1,mx5,i,i,a[i],i);
    }
    sc(m);
    while(m--)
    {
        ll check,l,r,k,x,v;
        sc(check);
        if(check==1)
        {
            sc(v);
            a[++n]=v;
            update_add(1,1,mx5,n,n,v,n);
        }
        else if(check==2)
        {
            update_delete(1,1,mx5,n,n,a[n],n);
            a[n--]=0;
        }
        else
        {
            ll L,R;
            sc(L),sc(R),sc(k);
            k=(R-L+1)-k+1;
            l=1,r=mx9;
```



```
            ll f=0;
            while(l<=r)
            {
                if(l==r)
                {
                    f++;
                    if(f==2)break;
                }
                ll mid=(l+r)/2;
                if(quary(1,1,mx5,L,R,mid,mx18)>=k) r=mid;

                else l=mid+1;
            }

            printf("%lld\n",l);

        }

    }

}
3.segment_tree
void build(int u,int b,int e)
{
    if(b==e)
    {
        tr[u]=a[b];
        return;
    }
    ll mid=(b+e)/2;
    build(2*u,b,mid);
```



```
    build(2*u+1,mid+1,e);
    tr[u]=min(tr[(2*u)],tr[(2*u)+1]);
}
ll quary(int u,int b,int e,int i,int j)
{
    if(e<i or b>j)return mx18;
    else if(b>=i and e<=j)
    {
        return tr[u];
    }
    int   mid=(b+e)/2;
    ll  left=quary(2*u,b,mid,i,j);
    ll right=quary((2*u)+1,mid+1,e,i,j);
    return  min(left,right);
}
ll bs(ll l,ll r,ll value)
{
    ll left=l;
    ll f=0,ans=-1;
    while(l<=r)
    {
        if(l==r)
        {
            f++;
            if(f==2)break;
        }
        ll mid=(l+r)/2;
        if(quary(1,1,n,left,mid)<value)
        {
            r=mid;
            ans=mid;
```



```
        }
        else l=mid+1;
    }
    return ans;
}

void input()
{
    ll m,val,ans=0,size=0;
    cin>>n>>m;
    for(int i=1; i<=n; i++)cin>>a[i];
    build(1,1,n);
    while(m--)
    {
        cin>>x;
        ll pre_id=bs(1,x-1,a[x]);
        //(1.....x-1 indx er modde x theke sob cheyere
dure kun indx a[x] theke chuto
        ll suf_id=bs(x+1,n,a[x]);
        //(x+1 ..........n indx er modde sobh theke kache
kun indx a[i] theke chuto
        if(pre_id==-1)cout<<"age nai cuto value"<<endl;
        else cout<<pre_id<<endl;
        if(suf_id==-1)cout<<"pore nai cuto value"<<endl;
        else cout<<suf_id<<endl;
    }
}
4.Optimize_Seive

const int N=mx8;
bitset<N>visited;
```

```
int prime[N],sz=0;
void seive()
{
    for(ll i=4;i<N;i+=2)visited[i]=1;
    for(ll i=3; i*i<N; i+=2)
    {
        if(visited[i])continue;
        for(ll j=i*i; j<N; j+=(2*i))
        {
            visited[j]=1;
        }
    }
    prime[0]=2;
    int cnt=1;
    for(int i=3;i<N;i+=2)
    {
      if(!visited[i])prime[cnt++]=i;
    }
    sz=cnt;
}
5.prime power of All possible divisor generator
vector< pair<ll,ll>>pr;
void f(int i, ll x)
{
    if(i == pr.size())
    {
        cout<<x<<" ";
        return ;
    }
    int p=pr[i].second;
    ll y=1;
```



```
    for(int j=0 ; j<=p ; j++)
    {
        f(i+1, x*y);
        y*=pr[i].first;
    }
}
void input()
{
    f(0,1);
}
6.Euler totient phi table

void pre_calculation()
{
    phi[1] = 1 ;
    for(int i=2; i<=1000000; i++)
    {
        if(phi[i]==0)
        {
            phi[i] = i-1 ;
            for(int j=i+i; j<=1000000; j+=i)
            {
                if(phi[j]==0) phi[j] = j ;
                phi[j] = phi[j] - phi[j]/i ;
            }
        }
    }
    for(int i=1; i<=1000000; i++)
    {
        for(int j=i; j<=1000000; j+=i)
        {
```



```
            S[j] = S[j] + (ll) (i*phi[i]);
        }
    }
}
6.nCr , mod is not prime

ll fac[2*mx7],smallprime[2*mx7],largeprime[2*mx7];
ll big_mod(ll b,ll p,ll m )
{
    ll res=1;
    while(p!=0)
    {
        if(p&1)res=(res*b)%m;
        b=(b*b)%m;
        p=p>>1;
    }
    res=(res)%m;
    return res;
}
void pre()
{
  //mod=103003811=103*1000037
    fac[0]=1;
    for(ll i=1; i<2*mx7; i++)
    {
        ll cnt=0;
        ll num=i;
        while(num%103==0)
        {
            cnt++;
            num/=103;
```



```
        }
        ll cnt2=0;
        while(num%1000037==0)
        {
            cnt2++;
            num/=1000037 ;
        }
        fac[i]=(fac[i-1]*num)%mod;
        smallprime[i]=smallprime[i-1]+cnt;
        largeprime[i]=largeprime[i-1]+cnt2;
    }
}
ll ext_euclid(ll a, ll b)
{
    ll q, ps=1, s=0, pt=0, t=1, r;
        if(b==0) while(1);
    while(a%b != 0)
    {
        q = a/b; r = a-q*b;
        ll tmps=s, tmpt=t;
        s = ps-q*s, t = pt-q*t;
        ps = tmps, pt = tmpt;
        a=b; b=r;
        if(b==0) while(1);
    }
    return (t+mod)%mod;
}
ll nCr(ll n,ll r)
{

    if(r>n)return 0;
```

```
    if(r==0)return 1;
    if(n<=0)return 0;
    if(r<0)return 0;

    ll res=(fac[n]);
    ll temp=fac[n-r]*fac[r];
    temp=ext_euclid(mod,temp);
    res=(res*temp)%mod;

    ll smlprime=smallprime[n]-smallprime[n-r]-smallprime[r];
    ll lrgprime=largeprime[n]-largeprime[n-r]-largeprime[r];

    res=(res*big_mod(103,smlprime,mod))%mod;
    res=(res*big_mod(1000037,lrgprime,mod))%mod;
    return res;
}
void input()
{
    ll n,r,ans=0;
    sc2(n,r);
    ans=nCr(n,r);
    printf("%lld\n",ans);

}
7.Ordered set
#include<bits/stdc++.h>
#include<ext/pb_ds/assoc_container.hpp>
using namespace std;
#include<ext/pb_ds/tree_policy.hpp>
```



```
using namespace __gnu_pbds;
typedef tree<int,null_type,less<int>,rb_tree_tag,
        tree_order_statistics_node_update>ordered_set;
ordered_set st;
void input()
{
    int  n,m,ans=0,sum=0;
    sci(n);
    for(int i=0;i<n;i++)
    {
        cin>>m;
        st.insert(m);
    }
     //n=10......0 1 4 5  6 8 9 10 11 15
     cout<<*(st.find_by_order(2))<<endl;//output=3;
     cout<<st.order_of_key(4)<<endl;
     //strictly less then 4 number of element output 2 ta
     int r=7,l=4;
     //upper_bound(r)-lower_bound(l)
      cout<<st.order_of_key(r+1)- st.order_of_key(l)<<endl;
      int x=2;
      if(st.find(x)!=st.end())
      {
          st.erase(st.find(x));//value x delete
      }
}
8.LCS
char in[1000][1000];
int lcs[1000][1000];
void  print(string A,int i,int j)
```



```
{
    if(i==0||j==0)
    {
        return ;
    }
    if(in[i][j]=='D')
    {
        print(A,i-1,j-1);
        cout<<A[i-1];
    }
    else if(in[i][j]=='U')
    {
        print(A,i-1,j);
    }
    else print(A,i,j-1);
}
int lcs_length(string A,string B)
{
    int m,n,i,j,k;
    m=A.size();
    n=B.size();
    for(i=0; i<=m; i++)
    {
        for(j=0; j<=n; j++)
        {
            if(i==0||j==0)
            {
                lcs[i][j]=0;
            }
            else  if(A[i-1]==B[j-1])
            {
```



```
                lcs[i][j]=1+lcs[i-1][j-1];
                in[i][j]='D';
            }
            else if(lcs[i-1][j]>=lcs[i][j-1])
            {
                lcs[i][j]=lcs[i-1][j];
                in[i][j]='U';
            }
            else
            {
                lcs[i][j]=lcs[i][j-1];
                in[i][j]='L';
            }
        }
    }
    return lcs[m][n];
}
9.Rod Cutting
void rod_cutting(int n,int price[])
{
 int tab[n+1][n+1];
 for(int i=0; i<=n; i++)
 {
    for(int j=0; j<=n; j++)
    {
      if(i==0 or j==0)tab[i][j]=0;
      else if(j<i) tab[i][j] = tab[i-1][j];
      else tab[i][j] = max(tab[i-1][j], price[i]+tab[i][j-i]);
     }
   }
```

```
    int i = n, j = n;
    vector<int> cut_point;
    while(i> 0 and j>0)
    {
        if(tab[i][j] == tab[i-1][j]) i--;
        else
        {
            cut_point.push_back(i);
            j = j-i;
        }
    }
}

10.Knapsack
ll knapSack( ll W, ll wt[], ll p[], ll n)
{
   ll i, w;
   ll k[n+1][W+1];
   for (i = 0; i <= n; i++)
   {
    for (w = 0; w <= W; w++)
     {
       if (i==0 || w==0)k[i][w] = 0;
       else if (wt[i-1] <= w)
       {
         k[i][w] = max((p[i-1] + k[i-1][w-wt[i-1]]), k[i-
1][w]);
       }
        else k[i][w] = k[i-1][w];
    }
 }
```



```
    return k[n][W];
}

11. Trie with Bit
(Binary trie all subrray xor sum
    with xor k in maximum,minimum(spoj XORX))
ll to[mx6][2],to_node=1,cs=0;
void trie_add_string(bitset<32> s)
{
    ll cur=1;
    for(int i=31; i>=0; i--)

        int  ch=s[i];
        if(!to[cur][ch])to[cur][ch]=++to_node;
        cur=to[cur][ch];
    }
}
ll trie_quary_max(bitset<32> s)
{
    ll cur=1,ans=0;
    ll d=1<<30;
    d*=2LL;
    for(int i=31; i>=0; i--)
    {
        int ch=s[i];
        if(to[cur][ch^1])
        {
            cur=to[cur][ch^1];
            ans+=d;
        }
        else cur=to[cur][ch];
```



```
        d/=2;

    }
    return ans;
}
ll trie_quary_min(bitset<32> s)
{
    ll cur=1,ans=0;
    ll d=1<<30;
    d*=2LL;
    for(int i=31; i>=0; i--)
    {
        int ch=s[i];
        if(to[cur][ch] )
        {
            cur=to[cur][ch];
        }
        else {
                ans+=d;
                cur=to[cur][ch^1];
        }
        d/=2;
    }
    return ans;
}

void input()
{
    ll n,l,r,res=0;
    sc(n);
    sc(r);
```



```
    bitset<32>ma(0);
    trie_add_string(ma);
    ll mx=0,pre=0;
    ll mn=mx18;

    ll ans=0;
    for(int i=0;i<n;i++){

    sc(l);
    pre^=l;

    ll d=pre^r;
    bitset<32>ma(d);
    mx=max(mx,trie_quary_max(ma));
    bitset<32>mb(pre);
    trie_add_string(mb);
    }
    printf("%lld\n",mx^r);
}
12. Trie template
void trie_add_string(string s)
{
    int cur=1;
    cnt[cur]++;
    for(int i=0; i<s.size(); i++)
    {
        int  ch=s[i]-'a';
        if(!to[cur][ch])to[cur][ch]=++to_node;
        cur=to[cur][ch];
        cnt[cur]++;
    }
```

```
    track[cur]++;
}
int trie_quary(string s)
{
    int cur=1;
    for(int i=0; i<s.size(); i++)
    {
        int ch=s[i]-'a';
        if(!to[cur][ch])return cnt[cur];
        cur=to[cur][ch];
    }
    return cnt[cur];
}
13.KMP (prefix occurs number of time)
ll cnt[mx6],to=0;
vector<ll> kmp_prefix_fun(string s)
{
    ll n=s.size();
    vector<ll>pi(n);
    //pi[0]=0;
    for(ll i=1; i<n; i++)
    {
        ll j=pi[i-1];
        while(j>0 and s[i]!=s[j])j=pi[j-1];
        if(s[i]==s[j])++j;
        pi[i]=j;
        cnt[j]++;
    }
    return pi;
}
int t=1;
```



```
void input()
{
   string a,b;
   cin>>a;
   vector<ll> v=kmp_prefix_fun(a);
   ll n=a.size();
   for(int i=n;i>=1;i--)
   {
       cnt[v[i-1]]+=cnt[i];
   }
  vector<ll>vec;
   while(n)
   {
       vec.pb(n);
       n=v[n-1];
   }
  ll sz=vec.size();
  cout<<sz<<endl;
   for( int i=sz-1;i>=0;i--)
   {
       cout<<vec[i]<<" "<<cnt[vec[i]]+1<<endl;
   }
}
14.Hashing Template
#include <bits/stdc++.h>
#define ff first
#define ss second
#define mp make_pair
using namespace std;
typedef long long LL;
typedef pair<LL, LL> PLL;
```



```
const PLL M=mp(1e9+7, 1e9+9);   ///Should be large primes
const LL base=347;              ///Should be a prime
larger than highest value
const int N = 1e6+7;            ///Highest length of
string

ostream& operator<<(ostream& os, PLL hash)
{
    return os<<"("<<hash.ff<<", "<<hash.ss<<")";
}
PLL operator+ (PLL a, LL x)
{
    return mp(a.ff + x, a.ss + x);
}
PLL operator- (PLL a, LL x)
{
    return mp(a.ff - x, a.ss - x);
}
PLL operator* (PLL a, LL x)
{
    return mp(a.ff * x, a.ss * x);
}
PLL operator+ (PLL a, PLL x)
{
    return mp(a.ff + x.ff, a.ss + x.ss);
}
PLL operator- (PLL a, PLL x)
{
    return mp(a.ff - x.ff, a.ss - x.ss);
}
PLL operator* (PLL a, PLL x)
```



```
{
    return mp(a.ff * x.ff, a.ss * x.ss);
}
PLL operator% (PLL a, PLL m)
{
    return mp(a.ff % m.ff, a.ss % m.ss);
}
PLL power (PLL a, LL p)
{
    if (p==0)   return mp(1,1);
    PLL ans = power(a, p/2);
    ans = (ans * ans)%M;
    if (p%2)    ans = (ans*a)%M;
    return ans;
}
///Magic!!!!!!
PLL inverse(PLL a)
{
    return power(a, (M.ff-1)*(M.ss-1)-1);
}
PLL pb[N];      ///powers of base mod M
PLL invb;
///Call pre before everything
void hashPre()
{
    pb[0] = mp(1,1);
    for (int i=1; i<N; i++)
        pb[i] = (pb[i-1] * base)%M;
    invb = inverse(pb[1]);
}
///Calculates Hash of a string
```

```
PLL Hash (string s)
{
    PLL ans = mp(0,0);
    for (int i=0; i<s.size(); i++)
        ans=(ans*base + s[i])%M;
    return ans;
}
///appends c to string
PLL append(PLL cur, char c)
{
    return (cur*base + c)%M;
}
///prepends c to string with size k
PLL prepend(PLL cur, int k, char c)
{
    return (pb[k]*c + cur)%M;
}

///replaces the i-th (0-indexed) character from right from a to b;
PLL replace(PLL cur, int i, char a, char b)
{
    cur = (cur + pb[i] * (b-a))%M;
    return (cur + M)%M;
}
///Erases c from the back of the string
PLL pop_back(PLL hash, char c)
{
    return (((hash-c)*invb)%M+M)%M;
}
///Erases c from front of the string with size len
```



```
PLL pop_front(PLL hash, int len, char c)
{
    return ((hash - pb[len-1]*c)%M+M)%M;
}

///concatenates two strings where length of the right is k
PLL concat(PLL left, PLL right, int k)
{
    return (left*pb[k] + right)%M;
}
///Calculates hash of string with size len repeated cnt times
///This is O(log n). For O(1), pre-calculate inverses
PLL repeat(PLL hash, int len, LL cnt)
{
    PLL mul = (pb[len*cnt] - 1) * inverse(pb[len]-1);
    mul = (mul%M+M)%M;
    PLL ans = (hash*mul)%M;

    if (pb[len].ff == 1)    ans.ff = hash.ff*cnt;
    if (pb[len].ss == 1)    ans.ss = hash.ss*cnt;
    return ans;
}
///Calculates hashes of all prefixes of s including empty prefix
vector<PLL> hashList(string &s)
{
    int n = s.size();
    vector<PLL> ans(n+1);
    ans[0] = mp(0,0);

```



```
    for (int i=1; i<=n; i++)
        ans[i] = (ans[i-1] * base + s[i-1])%M;
    return ans;
}
///Calculates hash of substring s[l..r] (1 indexed)
PLL substringHash(const vector<PLL> &hashlist, int l, int r)
{
    int len = (r-l+1);
    return ((hashlist[r] - hashlist[l-1]*pb[len])%M+M)%M;
}
vector<PLL> ha,hb;
int l1,r1,n;
bool ok(int mid)
{
    map<PLL,int>ma;
    for(int i=0; i<n; i++)
    {
        if(i+mid-1<n)
        {
            PLL  d=substringHash(hb,i+1,i+1+mid-1);
            ma[d]++;
        }
        else break;
    }
    for(int j=0; j<n; j++)
    {
        if(j+mid-1<n)
        {
            PLL d1=substringHash(ha,j+1,j+1+mid-1);
            if(ma[d1])
```



```
            {
                l1=j;
                r1=j+mid-1;
                return 1;
            }
        }
        else break;
    }
    return 0;
}
char buffer[N];
int main()
{
    hashPre();
    int t;
    t=1;
    for (int cs=1; cs<=t; ++cs)
    {
        string a,b;
        cin>>n>>a>>b;
        int na = a.size(), nb = b.size();
        hb=hashList(b);
        ha = hashList(a);
        int l=0,r=nb+1,f=0;
        while(l<=r)
        {
            if(l==r)
            {
                f++;
                if(f==2)break;
            }
```

```
            int mid=(l+r+1)/2;
            if(ok(mid))l=mid;
            else r=mid-1;
        }
        for(int i=l1; i<=r1; i++)
        {
            cout<<a[i];
        }
    }
}


14.All posible increasing sequence count;
ll mn[400005],a[400005];
map<ll,ll>ma;
vector<ll>v;
void setall2()
{
    ma.clear();
    v.clear();
    memset(mn,0,sizeof(mn));
}
void build(ll u,ll b,ll e)
{
    if(b==e)
    {
        mn[u]=0;
        return ;
    }
    ll mid=(b+e)/2;
    build(2*u,b,mid);
```



```
    build((2*u)+1,mid+1,e);
    mn[u]=mn[2*u]+mn[(2*u) +1];
}
ll quary (ll u,ll b,ll e,ll i,ll j)
{
    if(e<i or b>j)return 0;
    if(b>=i && e<=j)return mn[u];
    ll mid=(b+e)/2;
    ll l_mn=quary(2*u,b,mid,i,j);
    ll r_mn=quary(2*u+1,mid+1,e,i,j);
    return ((l_mn+r_mn)%mod+mod)%mod;

}
void update(ll  u,ll  b,ll  e,ll  l,ll  x)
{
    if( l>e or l<b)
    {
        return ;
    }
    else if(l<=b && e<=l)
    {
        mn[u]=((mn[u]+x)%mod+mod)%mod;
        return ;
    }
    ll  mid=(b+e)/2;
    update(2*u,b,mid,l,x);
    update(2*u+1,mid+1,e,l,x);
    mn[u]=((mn[2*u]+mn[(2*u)+1])%mod+mod)%mod;
}
void setall()
{
```



```

    sort(v.begin(),v.end());
    v.erase(unique(v.begin(),v.end()),v.end());
    for(i=0; i<v.size(); i++)
    {
        ma[v[i]]=i+1;
    }
build(1,1,v.size());
}
void solve()
{

    for(i=0; i<n; i++)
    {
        ll s=quary(1,1,v.size(),1,ma[a[i]]-1)+1;
        update(1,1,v.size(),ma[a[i]],s);
    }
    printf("Case %lld: %lld\n",++f,mn[1]);

}
void input()
{
    sc(t);
    while(t--)
    {
        setall2();
        sc(n);
        for(i=0; i<n; i++)
        {
            sc(a[i]);
            v.pb(a[i]);
```



```
        }
        setall();
        solve();
    }

}
15.All possible  subarray xor sum
int main()
{
    cin>>t;
    while(t--)
    {
        cin>>n;
        long long int b[n+1];
        b[0]=0;
        sum=0;
        long long int a[33]= {0};
        for(i=1; i<=n; i++)
        {   cin>>m;
            b[i]=b[i-1]^m;
        }

        k=1;
        for(i=0; i<=32; i++)
        {
            c=0;
            for(j=1; j<=n; j++)
            {
                if(b[j]&(1LL<<i))c++;

            }
```

```
            sum+=(c*(n+1-c)*k);
            k=k<<1;
        }
        cout<<sum<<endl;
    }
}
16.Bigmod()
ll big_mod(ll b,ll p,ll m )
{
    ll res=1;
    while(p!=0)
    {
        if(p&1)res=(res*b)%m;
        b=(b*b)%m;
        p=p>>1;
    }
    res=(res)%m;
    return res;
}
17.ALL pair lCM sum

ll cnt[mx6],b[mx6],exact[mx6];
void seive()
{
    for(ll i=1; i<mx6; i++)
    {
        for(ll j=i; j<mx6; j+=i)
        {
            b[i]+=(j*cnt[j]);
        }
        b[i]%=mod;
```



```
    }
}

void input()
{
    ll n,m,ans;
    cin>>n;
    for(ll i=0; i<n; i++)
    {
        cin>>m;
        cnt[m]++;
    }
    seive();
    for(ll i=1000000; i>=1; i--)
    {
        exact[i]=(b[i]*b[i])%mod;
        for(int j=i+i; j<=1000000; j+=i)
        {
            exact[i]=(exact[i]-exact[j]);
            if(exact[i]<0)exact[i]+=mod;

        }
        ans=(ans+exact[i]*big_mod(i,mod-2ll,mod))%mod;
        ans=(ans-(i*cnt[i]))%mod;
        if(ans<0)ans+=mod;
    }
    ans=(ans*big_mod(2,mod-2,mod))%mod;
    ans=(ans+mod)%mod;

    printf("%lld\n",ans);

```



```
}
//Exact Gcd g
void input()
{
    for(ll i=1000000; i>=1; i--)
    {
         if(b[i]==0)d=0;
         else exact[i]=d;
        for(int j=i+i; j<=1000000; j+=i)
        {
            exact[i]=(exact[i]-exact[j]);
        }
    }

    printf("%lld\n",ecaxt[G]);

}
//String stream
//string to num
int x;
string a="375834";
stringstream ss(a);
ss>>x;
cout<<x<<endl;
//number to string
int n=45;
stringstream ss;
ss<<n;
string a=ss.str();
cout<<a<<endl;

```



```
ll count word(string a)
{
    stringstream ss(a);
    string word;
    while(ss>>word)cnt++;
    return cout<<cnt<<endl;
}

//Catalan Number

Cn=(2n!)/(((n+1)!) * (n!));
n=0,1,2,3,....
cn=1,1,2,5,14,42,132,429,1430....

//Increasing sequence
ll b[mx6],a[mx6],i;
void bs(ll m)
{
    ll l=0,r=v.size()-1;
    while(l<r)
    {
        ll mid=(l+r-1)/2;
        if(v[mid]>=m)r=mid;
        else l=mid+1;
    }
    if(v[r]==m)
    {
        b[i]=r;
    }
    else if(v[r]<m)
    {
```

```
        v.pb(m);
        b[i]=r+1;
    }
    else
    {
       v[r]=min(m,v[r]);
       b[i]=r;
    }
}
void solve()
{
    ll n,ma=0;
    cin>>n;
    for(i=0;i<n;i++)
    {
        cin>>a[i];
    }
    v.pb(a[0]);
    for( i=1;i<n;i++)
    {
        bs(a[i]);
    }
    for(i=0;i<n;i++)ma=max(ma,b[i]);
    cout<<ma+1<<endl;
}

//COMSOD(n)=i to n sumof( sum of number of divisor)

for(int i=2;i*i<=n;i++)
{
    j=n/i;
```



```
    ans+=((i+j)*(j-i+1))/2;
    ans+=i*(j-i);
}
cout<<ans<<endl;

//Multiset
int main()
{
    multiset<int> st//increasing order
    for (itr = gquiz1.begin(); itr != gquiz1.end(); ++itr)
{
        cout << *itr << " ";
    }
    cout << endl;
    // remove all elements up to element
    // with value 30 in gquiz2
    gquiz2.erase(gquiz2.begin(), gquiz2.find(30));
    // remove all elements with value 50 in gquiz2
    int num;
    num = gquiz2.erase(50);
    // lower bound and upper bound for multiset gquiz1
    cout << "\ngquiz1.lower_bound(40) : \n"
         << *gquiz1.lower_bound(40) << endl;
    cout << "gquiz1.upper_bound(40) : \n"
         << *gquiz1.upper_bound(40) << endl;

    return 0;
}

//Array Range check
```



```
ca=max(a,l);
cb=min(b,r);
if(ca>cb)cout<<out of range;
else cout<<in range;

//All possible Permutation a[i]!=i
int count(int n)
{
    if(n==1)return 0;
    if(n==2)return 1;
    return (n-2)*((count(n-1)+count(n-2));
}

//NEXT elecent after delete using Bitset

bitset<100000<bit;
bit.flip();
1111111111111111
bit[1]=0;
bit[4]=
10110111111
bit._Find_next(b-1);
// b index er soman or pore  kothay bit on  ache

//Locas Combination bignumber nCr
mod=99999997;
llu Locascombination(llu n,llu r)
{
    if(r==0)return 1;
    llu ni=n%mod,ri=r%mod;
    if(ni<ri)return 0;
```



```
    return (((fac[ni]*big_mod(fac[ri],mod-
2,mod)%mod)*bigmod(fac[ni-ri],mod-
2,mod))%mod)*Locascombination(n/mod,r/mod))%mod);
}

//Combinatorics star and vars
1. x1+x2+x3+x4...+xk=n;
xi>=1;
number of way=(n-1)C (k-1)

2.
x1+x2+x3+x4...+xk<=n;
xi>=1;
solution :
x1+x2+x3+x4...+xk  + m <=n;
number of way=summation of (indx m=0 to n-k)   (n-m-1)C(k-
1);

//Degree of Time

ans=abs((11*m -60*h)/2);
//Template
#include<bits/stdc++.h>
using namespace std;
#define TT ios::sync_with_stdio(false);
cin.tie(0);cout.tie(0)
#define      ll                  long long int
#define      ull                 unsigned long long int
#define      vi                  vector<int>
#define      vc                  vector<char>
#define      vs                  vector<string>
```

```
#define       vll                 vector<long long int>
#define       vp                  vector<  pair<int,int>
>
#define       pb                  push_back
#define       pob                 pop_back
#define       pll                 pair<long long  int,
long long int>
#define       F                   first
#define       S                   second
#define       sc(x)               scanf("%lld",&x)
#define       sci(x)               scanf("%d",&x)
#define       sc2(x,y)            scanf("%lld
%lld",&x,&y)
#define       pf                  printf
#define       min3(a,b,c)         min(a,b<c?b:c)
#define       max3(a,b,c)         max(a,b>c?b:c)
#define       all(v)              v.begin(), v.end()
#define       rall(v)              v.rbegin(), v.rend()
///============ CONSTANT ==============///
#define mx18  100000000000000000
#define mx9   1000000007
#define mx8   100000007
#define mx7   10000006
#define mx6   1000056
#define mx5   200005
#define mx4   10005
#define inf   1<<30
#define eps   1e-9
#define mod   mx9
ll dx[]= {1,-1,0,0};
ll dy[]= {0,0,1,-1};
```



```
long double PI = acosl(-1);
///============== Debugging
============================///
#define       debug(x) cerr << #x << " = " << x << endl
#define       debug2(x, y)            cerr << #x << ": "
<< x << "  " << #y << ": " << y << endl;
#define       debug3(x, y, z)         cerr << #x << ": "
<< x << "  " << #y << ": " << y << "  " << #z << ": " << z
<< endl;
#define       debug4(a, b, c, d)      cerr << #a << ": "
<< a << "  " << #b << ": " << b << "  " << #c << ": " << c
<< " | " << #d << ": " << d << endl;
///====================Bitmask==================///
//ordered_set st;
//int Set(int N,int pos){return N=N | (1<<pos);}
//int reset(int N,int pos){return N= N & ~(1<<pos);}
//bool check(int N,int pos){return (bool)(N & (1<<pos));}
```