```
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();
}
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 ;
            }
        }
    }

}

/*
formula
lcm(1,n)+lcm(2,n)+lcm(3,n)+.....lcm(n,n)
SUM=(n*(∑(ϕ(d)×d)+1))/2; here d is divisor of n

for(int i=1; i<=1000000; i++)
   for(int j=i; j<=1000000; j+=i)
       S[j] = S[j] + (ll) (i*phi[i]);
         cin>>n;
         res=n*(s[n]+1);
         res/=2;
*/

// here  lcm(1,1)    +         lcm(1,2)+lcm(2,2)
      +       lcm(1,3)+lcm(2,3)+lcm(3,3) +
```



```
.............
Pseudocode code:
/*for( int i = 1; i <= n; i++ )
     for( int j = i + 1; j <= n; j++ )
         res += lcm(i, j);*/
for(int i=1; i<=1000000; i++)
   for(int j=i; j<=1000000; j+=i)
       S[j] = S[j] + (ll) (i*phi[i]);

    res[1]=1;
    res[1]-=1;
    for(ull i=2; i<3*mx6; i++)
    {
     ull tem=(1ull*i*((s[i]-1)/2ull));//
lcm(1,n)+lcm(2,n)+lcm(3,n)....lcm(n-1,n)  s[i]-1 k
age 2 dara vag then i er sate gun
        res[i]+=res[i-1]+tem;
    }

#GCD(i,n)  1<=i<=n-1
Pseudocode code:
/*for( int i = 1; i <= n; i++ )
   for( int j = i + 1; j <= n; j++ )
         res += lcm(i, j);*/

for(int i=1; i<=1000000; i++)
  for(int j=i; j<=1000000; j+=i)
     S[j] = S[j] + (ll) (i*phi[j/i]);
```

```
    sum[0]=0;
 for(int i=1;i<=n;i++)
    sum[i]=s[i]+sum[i-1];

#L to R  all minimum lcm
sc(m);sc(n);
    ll  sz=sqrt(n);
    for(int i=1;i<=sz;i++)
    {
        ll d=(m+i-1)/i;
        d*=i;
        if(d>=m and d<=n and (d+i)<=n)
        {
            ans=min(ans,1ull*d*(d+i)/i);
        }
        d=(m+i-1)/i;

         if(d*i>=m and d*i<=n and d*i +d<=n)
        {
            ans=min(ans,1ull*i*d*(i+1));
        }

    }
    printf("%llu\n",ans);

#sum of lcm(i,j)=n or lcm of all divisors is n.cpp
Pseudocode code:
```



```
/*for( int i = 1; i <= n; i++ )
   for( int j = i ; j <= n; j++ )
         if(lcm(i,j)==n)cnt++;
*/
 for(int i=0; i<sz and prime[i]*prime[i]<=n; i++)
      { if(n%prime[i]==0)
            ll d=prime[i],cnt=0;
            while(n%d==0)
            { n/=d;cnt++;
            }
            ans*=(2*cnt +1);
        }
    if(n>1)ans*=(2 +1);
    ans++; ans/=2;
cout<<ans<<endl;

17.ALL pair lCM sum
/*3
2 4 6
lcm(2,4)+lcm(2,6)+lcm(4,6)=4+6+12=22.*/
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]);

}
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;
    }
}
# segment tree index find

ll quary(int u,int b,int e,int val)
{
   if(b==e)
    {
        return b;
    }
     int   mid=(b+e)/2;
      if(tr[2*u]>=val)return quary(2*u,b,mid,val);
       return quary((2*u)+1,mid+1,e,val-tr[2*u]);
}

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(query(1,1,mx5,L,R,mid,mx18)>=k)r=mid;

                else l=mid+1;
            }

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



```
#Lazy with segment tree

ll tr[4*100005],lazy[4*100005];
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]=(tr[(2*u)]+tr[(2*u)+1]);

}
ll quary(int u,int b,int e,int i,int j)
{
    if(lazy[u])
    {
        tr[u] = (e-b+1)-tr[u];
        if(b != e)
        {
            lazy[2*u] ^= 1;
            lazy[2*u+1] ^=1;
        }
        lazy[u] = 0;
    }
```

```
    if(e<i or b>j)return 0;
    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  (left+right);
}
void  update (int u,int b,int e,int i,int j,ll x)
{

    if(lazy[u])
    {
        tr[u] = (e-b+1)-tr[u];
        if(b != e)
        {
            lazy[2*u] ^= lazy[u];
            lazy[2*u+1] ^= lazy[u];
        }
        lazy[u] = 0;
    }
    if(e<i or b>j)return;
    else if(b>=i and e<=j)
        {

        tr[u] = (e-b+1)-tr[u];
        if(b != e)
        {
            lazy[2*u] ^= 1;
            lazy[2*u+1] ^= 1;
        }
        lazy[u] = 0;
         return;
        }
        int mid=(b+e)/2;
        update(2*u,b,mid,i,j,x);
        update((2*u)+1,mid+1,e,i,j,x);
        tr[u]=(tr[2*u]+tr[2*u +1]);
}
#Array range in range or out of range
fun(a,b,l,r)
{
ca=max(a,l);
cb=min(b,r);
if(ca>cb)out of range
else in range

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);

}
#Catalan Number

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

```
652.       int x=2;
653.       if(st.find(x)!=st.end())
654.       {
655.           st.erase(st.find(x));//value x delete
656.       }
657. }
658. 8.LCS
659. char in[1000][1000];
660. int lcs[1000][1000];
661. void  print(string A,int i,int j)
662. {
663.     if(i==0||j==0)
664.     {
665.         return ;
666.     }
667.     if(in[i][j]=='D')
668.     {
669.         print(A,i-1,j-1);
670.         cout<<A[i-1];
671.     }
672.     else if(in[i][j]=='U')
673.     {
674.         print(A,i-1,j);
675.     }
676.     else print(A,i,j-1);
677. }
678. int lcs_length(string A,string B)
679. {
```



```
680.     int m,n,i,j,k;
681.     m=A.size();
682.     n=B.size();
683.     for(i=0; i<=m; i++)
684.     {
685.         for(j=0; j<=n; j++)
686.         {
687.             if(i==0||j==0)
688.             {
689.                 lcs[i][j]=0;
690.             }
691.             else  if(A[i-1]==B[j-1])
692.             {
693.                 lcs[i][j]=1+lcs[i-1][j-1];
694.                 in[i][j]='D';
695.             }
696.             else if(lcs[i-1][j]>=lcs[i][j-1])
697.             {
698.                 lcs[i][j]=lcs[i-1][j];
699.                 in[i][j]='U';
700.             }
701.             else
702.             {
703.                 lcs[i][j]=lcs[i][j-1];
704.                 in[i][j]='L';
705.             }
706.         }
707.     }
```



```
708.     return lcs[m][n];
709. }
710. 9.Rod Cutting
711. void rod_cutting(int n,int price[])
712. {
713.  int tab[n+1][n+1];
714.  for(int i=0; i<=n; i++)
715.  {
716.     for(int j=0; j<=n; j++)
717.     {
718.       if(i==0 or j==0)tab[i][j]=0;
719.       else if(j<i) tab[i][j] = tab[i-1][j];
720.       else tab[i][j] = max(tab[i-1][j],
price[i]+tab[i][j-i]);
721.      }
722.    }
723.     int i = n, j = n;
724.     vector<int> cut_point;
725.     while(i> 0 and j>0)
726.     {
727.         if(tab[i][j] == tab[i-1][j]) i--;
728.         else
729.         {
730.             cut_point.push_back(i);
731.             j = j-i;
732.         }
733.     }
734. }
```



```
735.
736. 10.Knapsack
737. ll knapSack( ll W, ll wt[], ll p[], ll n)
738. {
739.    ll i, w;
740.    ll k[n+1][W+1];
741.    for (i = 0; i <= n; i++)
742.    {
743.     for (w = 0; w <= W; w++)
744.      {
745.        if (i==0 || w==0)k[i][w] = 0;
746.        else if (wt[i-1] <= w)
747.        {
748.          k[i][w] = max((p[i-1] + k[i-1][w-wt[i-
1]]), k[i-1][w]);
749.        }
750.         else k[i][w] = k[i-1][w];
751.     }
752.  }
753.     return k[n][W];
754. }
755.
756. 11. Trie with Bit
757. (Binary trie all subrray xor sum
758.     with xor k in maximum,minimum(spoj XORX))
759. ll to[mx6][2],to_node=1,cs=0;
760. void trie_add_string(bitset<32> s)
761. {
```



```
762.     ll cur=1;
763.     for(int i=31; i>=0; i--)
764.
765.         int  ch=s[i];
766.         if(!to[cur][ch])to[cur][ch]=++to_node;
767.         cur=to[cur][ch];
768.     }
769. }
770. ll trie_quary_max(bitset<32> s)
771. {
772.     ll cur=1,ans=0;
773.     ll d=1<<30;
774.     d*=2LL;
775.     for(int i=31; i>=0; i--)
776.     {
777.         int ch=s[i];
778.         if(to[cur][ch^1])
779.         {
780.             cur=to[cur][ch^1];
781.             ans+=d;
782.         }
783.         else cur=to[cur][ch];
784.         d/=2;
785.
786.     }
787.     return ans;
788. }
789. ll trie_quary_min(bitset<32> s)
```



```
790. {
791.     ll cur=1,ans=0;
792.     ll d=1<<30;
793.     d*=2LL;
794.     for(int i=31; i>=0; i--)
795.     {
796.         int ch=s[i];
797.         if(to[cur][ch] )
798.         {
799.             cur=to[cur][ch];
800.         }
801.         else {
802.                 ans+=d;
803.                 cur=to[cur][ch^1];
804.         }
805.         d/=2;
806.     }
807.     return ans;
808. }
809.
810. void input()
811. {
812.     ll n,l,r,res=0;
813.     sc(n);
814.     sc(r);
815.     bitset<32>ma(0);
816.     trie_add_string(ma);
817.     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;
}

//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;
}
//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);

20. 2D prefix sum

ll prefix[1050][1050],n,m;
void pre()
{
  for(int i=1;i<=n;i++)
   for(int j=1;j<=m;j++)
     prefix[i][j]=a[i][j]+prefix[i-1][j]+prefix[i]
[j-1]-prefix[i-1][j-1];
}

```



```
ll quary(ll x1,ll y1,ll x2,ll y2)
{
  return (prefix[x2][y2]-prefix[x1-1][y2]-
prefix[x2][y1-1]+prefix[x1-1][y1-1]);
}
# 2D difference Array

D[x][y]=a[x][y]+a[x-1][y-1]-a[x][y-1]-a[x-1][y];

a[x][y] = a[x][y-1] + a[x-1][y] - a[x-1][y-1] +
D[x][y];

# Fibonacci 1 to 10^18

map<long, long> F;
long  f(long n) {
    if (F.count(n)) return F[n];
    long k=n/2;
    if (n%2==0) { // n=2*k
        return F[n] = (f(k)*f(k) + f(k-1)*f(k-1)) %
M;
    } else { // n=2*k+1
        return F[n] = (f(k)*f(k+1) + f(k-1)*f(k)) %
M;
    }
}

```



```
21.Longest Palindromic Substring O(N) Manachers
Algorithm
int main() {
  int i, j, k, n, m;
  string s;
  cin >> s;
  n = s.size();
  vector<int> d1(n);  // maximum odd length
palindrome centered at i
  // here d1[i]=the palindrome has d1[i]-1 right
characters from i
  // e.g. for aba, d1[1]=2;
  for (int i = 0, l = 0, r = -1; i < n; i++) {
    int k = (i > r) ? 1 : min(d1[l + r - i], r -
i);
    while (0 <= i - k && i + k < n && s[i - k] ==
s[i + k]) {
      k++;
    }
    d1[i] = k--;
    if (i + k > r) {
      l = i - k;
      r = i + k;
    }
  }
  vector<int> d2(n);  // maximum even length
palindrome centered at i
```



```
  // here d2[i]=the palindrome has d2[i]-1 right
characters from i
  // e.g. for abba, d2[2]=2;
  for (int i = 0, l = 0, r = -1; i < n; i++) {
    int k = (i > r) ? 0 : min(d2[l + r - i + 1], r
- i + 1);
    while (0 <= i - k - 1 && i + k < n && s[i - k -
1] == s[i + k]) {
      k++;
    }
    d2[i] = k--;
    if (i + k > r) {
      l = i - k - 1;
      r = i + k ;
    }
  }
  for(i = 0; i < n; i++) cout << d1[i] << ' ';
  cout << endl;
  for(i = 0; i < n; i++) cout << d2[i] << ' ';
  cout << endl;
  // number of palindromes
  long long ans = 0;
  for(i = 0; i < n; i++) {
    ans += 1LL * d1[i];
    ans += 1LL * d2[i];
  }
  cout << ans << endl;
  return 0;
```

```
}
/*
aaaaabbbbaaaa
output:
1 2 3 2 1 1 2 2 1 1 2 2 1
0 1 2 2 1 0 1 6 1 0 1 2 1
39
*/
#Using Manacher Algorithm prefix +suffix maximum palinedrome
int main() {
   int t;
    cin>>t;
    while(t--)
    {
  int i, j, k, n, m;
  string s;
  cin >> s;
  n = s.size();
int st=0,end=n;
  for(int i=0;i<(n/2);i++)
  {
    if(s[i]==s[n-i-1])
    {
        st++;
        end--;
    }
    else break;
```



```
  }
  vector<int> d1(n);  // maximum odd length palindrome centered at i
  // here d1[i]=the palindrome has d1[i]-1 right characters from i
  // e.g. for aba, d1[1]=2;
  for (int i = st, l = st, r = st-1; i < end; i++) {
    int k = (i > r) ? 1 : min(d1[l + r - i], r - i);
    while (st<= i - k && i + k < end && s[i - k] == s[i + k]) {
      k++;
    }
    d1[i] = k--;
    if (i + k > r) {
      l = i - k;
      r = i + k;
    }
  }
  vector<int> d2(n);  // maximum even length palindrome centered at i
  // here d2[i]=the palindrome has d2[i]-1 right characters from i
  // e.g. for abba, d2[2]=2;
  for (int i = st, l = st, r = st-1; i < end; i++) {
```



```
    int k = (i > r) ? 0 : min(d2[l + r - i + 1], r - i + 1);
    while (st <= i - k - 1 && i + k < end && s[i - k - 1] == s[i + k]) {
      k++;
    }
    d2[i] = k--;
    if (i + k > r) {
      l = i - k - 1;
      r = i + k ;
    }
  }
int  odd=0,even=0,odd_id=0,even_id=0;
  for(int i=st;i<end;i++)
  {

      if(i+1==d1[i]+st)
      {
        if(odd<d1[i])
        {
            odd_id=i;
            odd=d1[i];
        }
      }
      if(i==(end-d1[i]))
      {
        if(odd<d1[i])
        {
```



```
            odd_id=i;
            odd=d1[i];
        }
      }
  }
//cout<<odd<<" "<<odd_id<<endl;
   for(int i=st;i<end;i++)
  {
      if(i==d2[i]+st)
      {
        if(even<d2[i])
        {
            even_id=i;
            even=d2[i];
        }

      }
      if(i==(end-d2[i]))
      {
        if(even<d2[i])
        {
            even_id=i;
            even=d2[i];
        }
      }
  }
   i=0;
  while(i<st){cout<<s[i];i++;}
```



```
  if(odd<=even)
  {
     for(int i=even_id-even;i<even_id+even;i++)cout<<s[i];
  }
else
{
     for(int i=odd_id-odd+1;i<odd_id+odd;i++)cout<<s[i];
}
   i=end;
  while(i<=n-1)
  {
    cout<<s[i];
    i++;
  }
  cout<<endl;
  // number of palindromes
 /* long long ans = 0;
  for(i = 0; i < n; i++) {
    ans += 1LL * d1[i];
    ans += 1LL * d2[i];
  }
   */
}
  return 0;
}
/*
```



```
aaaaabbbbaaaa
output:
1 2 3 2 1 1 2 2 1 1 2 2 1
0 1 2 2 1 0 1 6 1 0 1 2 1
39
*/
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==================///

//nth non square number
 ans(n)=n+floor(0.5+(double)sqrt(n));

//Basic counting
i.  (nc0)^2 +(nc1)^2+(nc2)^2+.....+(ncn)^2=(2nCn)
```

```
ii. (nc0)+((n+1)C1)+((n+2)c2)+..+((n+k)ck)=
(n+k+1)Ck
iii. 0.(nc0)+1.(nc1)+2.(nc2)+...+n(ncn)=n*(2^(n-1))
//Star & Bars
i.  x1+x2+x3+...xk=N
  ans=(N-1)c(k-1

 (All time we consider xi>=1)
 suppose x1+x2=3
  x1>=-5,x2>=2
  convert x1'=x1+6,x2'=x2-1

ii. x1+x2+x3.. <=N and xi>=1
  x1+x2+x3<=10
  we write x1+x2+x3+c<=10
  c=0,1,2,3,4,5,6,7;
       n-k  n-c-1
  ans= E       C     //E summation
       c=0      k-1
iii. 0<x1<x2<x3<x4<N
 d1=x1;d2=x2-x1,d3=x3-x2...
 consider di>=1
 d1+d2+d3+..+dk=N
     N-1
 formula=  C
       k-1
```



```
iv.n lenth er sequence a atleast 0 or 1 or 2 once
occur
  total=3^n
  badsequence=n*((2^n)-1);
  good=3^n -n*((2^n) -1)

//Series
i.   1+2+3+4.........n=(n*(n+1))/2
ii.  1^2+2^2+3^2..+n^2=(n*(n+1)*(n+2))/6
iii. 1^3+2^3+3^3..+n^3=((n*(n+1))/2)^2
iv.  4+7+10+......+n
     nth=a+(n-1)*d
     Sn=(n*(1st+nth))/2
     or Sn=(n*(2a+(n-1)*d))/2
v.   Gunotttor dara
     Sn=(a*(r^n -1))/r-1
     Sn=a/(1-r)
vi.  2+4+6+8.....+n=n*(n+1)
vii. 1+3+5+7+....+n=n^2
viii for(k=1 to n)sum+=(k*(k+1))
     formula=(n*(n+1)*(n+2))/3
ix.  for(k=1 to n)sum+=1/(k*(k+1))
     formula=n/(n+1)
x.   for(k=1 to n)sum+=(k*(k+1)*(k+2))
     formula=(n*(n+1)*(n+2)*(n+3))/4
xi.  for(k=1 to n)sum+=1/(k*(k+1)*(k+2))
     formula=(n*(n+3))/(4*(n+1)*(n+2))
XII. 2^2 +4^2+6^2+...+98^2
```



```
     nth=a+(n-1)*d
     98=2*(n-1)*2
     n=49
     for(1 to 49)=(2n)^2=4*n^2
formula=4*((n*(n+1)*(n+2))/6)
```