目录

[类欧几里得 2](#_Toc514256275)

[杜教BM板子(求递推式) 2](#_Toc514256276)

[并查集 整体维护 3](#_Toc514256277)

[Trie xor后小于limit最大值 4](#_Toc514256278)

[DP套DP 4](#_Toc514256279)

[虚树 6](#_Toc514256280)

[C(n,n) 7](#_Toc514256281)

[SPLAY\_NEW 7](#_Toc514256282)

[积性函数 10](#_Toc514256283)

类欧几里得

一定注意前面是a,后面是b,线段树一定要注意顺序

f(a,b,c,n)=sigma{(ai+b)/c}; (0->n)

g(a,b,c,n)=sigma{(ai+b)/c\*i}; (0->n)

h(a,b,c,n)=sigma{((ai+b)/c)^2}; (0->n)

let m=(a\*n+b)/c;

推导f:

a=0:

return b/c\*(n+1)

a>=c||b>=c:有一部分是规律的;

return (a/c)\*n(n+1)/2+(b/c)\*(n+1)+f(a%c,b%c,c,n)

else:直接算,这个东西是个梯形中的点数,反过来算就可以了

f(a,b,c,n)=∑i=0->n ∑j=0->m-1 [(ai+b)/c>=j+1]

f(a,b,c,n)=∑i=0->n ∑j=0->m-1 [ai>=cj+c−b]

f(a,b,c,n)=∑i=0->n ∑j=0->m-1 [ai>cj+c−b−1]

f(a,b,c,n)=∑i=0->n ∑j=0->m-1 [i>(cj+c−b−1)/a]

f(a,b,c,n)=∑j=0->m (n−(cj+c−b−1)/a)

f(a,b,c,n)=n\*m-f(c,c-b-1,a,m-1);

推导g:

a=0:

return b/c\*n(n+1)/2 (sigma的是i)

a>=c||b>=c:有一部分是规律的;

g(a,b,c,n)=(a/c)\*n(n+1)(2n+1)/6+(b/c)\*n(n+1)/2+g(a%c,b%c,c,n)

else:

g(a,b,c,n)=∑i=0->n i\*∑j=0->m [(ai+b)/c>=j]

g(a,b,c,n)=∑i=0->n i\*∑j=0->m-1 [i>(cj+c−b−1)/a]

然后把这个i放进去求和

g(a,b,c,n)=1/2\*∑j=0->m-1 (n+1+(cj+c−b−1)/a)\*(n−(cj+c−b−1)/a)

g(a,b,c,n)=1/2\*∑j=0->m-1 n(n+1)−(cj+c−b−1)/a−[(cj+c−b−1)/a]^2

g(a,b,c,n)=1/2\*[n(n+1)\*m−f(c,c−b−1,a,m−1)−h(c,c−b−1,a,m−1)]

推导h:

a=0:

return (b/c)^2\*(n+1) (sigma的是i)

a>=c||b>=c:有一部分是规律的;

h(a,b,c,n)=(a/c)^2\*n(n+1)(2n+1)/6+(b/c)^2\*(n+1)+(a/c)\*(b/c)\*n(n+1)

+h(a%c,b%c,c,n)+2\*(a/c)\*g(a%c,b%c,c,n)+2\*(b/c)\*f(a%c,b%c,c,n)

else:

n^2=2\*n(n+1)/2−n=2(∑i=0->n i)−n

有了思路我们来推h

h(a,b,c,n)=∑i=0->n (2(∑j=1->(ai+b)/c j)−(ai+b)/c)

可以想到交换主体。

h(a,b,c,n)=∑j=0->m-1 (j+1)\*∑i=0->n [(ai+b)/c>=j+1]−f(a,b,c,n)

h(a,b,c,n)=∑j=0->m-1 (j+1)\*∑i=0->n [i>(cj+c−b−1)/a]−f(a,b,c,n)

h(a,b,c,n)=∑j=0->m-1 (j+1)\*(n−(cj+c−b−1)/a)−f(a,b,c,n)

h(a,b,c,n)=n\*m(m+1)−2g(c,c−b−1,a,m−1)−2f(c,c−b−1,a,m−1)−f(a,b,c,n)

杜教BM板子(求递推式)

#include <cstdio>

#include <cstring>

#include <cmath>

#include <algorithm>

#include <vector>

#include <string>

#include <map>

#include <set>

#include <cassert>

using namespace std;

#define rep(i,a,n) for (int i=a;i<n;i++)

#define per(i,a,n) for (int i=n-1;i>=a;i--)

#define pb push\_back

#define mp make\_pair

#define all(x) (x).begin(),(x).end()

#define fi first

#define se second

#define SZ(x) ((int)(x).size())

typedef vector<int> VI;

typedef long long ll;

typedef pair<int,int> PII;

const ll mod=1000000007;

ll powmod(ll a,ll b) {ll res=1;a%=mod; assert(b>=0); for(;b;b>>=1){if(b&1)res=res\*a%mod;a=a\*a%mod;}return res;}

// head

int \_,n;

namespace linear\_seq {

const int N=10010;

ll res[N],base[N],\_c[N],\_md[N];

vector<int> Md;

void mul(ll \*a,ll \*b,int k) {

rep(i,0,k+k) \_c[i]=0;

rep(i,0,k) if (a[i]) rep(j,0,k) \_c[i+j]=(\_c[i+j]+a[i]\*b[j])%mod;

for (int i=k+k-1;i>=k;i--) if (\_c[i])

rep(j,0,SZ(Md)) \_c[i-k+Md[j]]=(\_c[i-k+Md[j]]-\_c[i]\*\_md[Md[j]])%mod;

rep(i,0,k) a[i]=\_c[i];

}

int solve(ll n,VI a,VI b) { // a 系数 b 初值 b[n+1]=a[0]\*b[n]+...

// printf("%d\n",SZ(b));

ll ans=0,pnt=0;

int k=SZ(a);

assert(SZ(a)==SZ(b));

rep(i,0,k) \_md[k-1-i]=-a[i];\_md[k]=1;

Md.clear();

rep(i,0,k) if (\_md[i]!=0) Md.push\_back(i);

rep(i,0,k) res[i]=base[i]=0;

res[0]=1;

while ((1ll<<pnt)<=n) pnt++;

for (int p=pnt;p>=0;p--) {

mul(res,res,k);

if ((n>>p)&1) {

for (int i=k-1;i>=0;i--) res[i+1]=res[i];res[0]=0;

rep(j,0,SZ(Md)) res[Md[j]]=(res[Md[j]]-res[k]\*\_md[Md[j]])%mod;

}

}

rep(i,0,k) ans=(ans+res[i]\*b[i])%mod;

if (ans<0) ans+=mod;

return ans;

}

VI BM(VI s) {

VI C(1,1),B(1,1);

int L=0,m=1,b=1;

rep(n,0,SZ(s)) {

ll d=0;

rep(i,0,L+1) d=(d+(ll)C[i]\*s[n-i])%mod;

if (d==0) ++m;

else if (2\*L<=n) {

VI T=C;

ll c=mod-d\*powmod(b,mod-2)%mod;

while (SZ(C)<SZ(B)+m) C.pb(0);

rep(i,0,SZ(B)) C[i+m]=(C[i+m]+c\*B[i])%mod;

L=n+1-L; B=T; b=d; m=1;

} else {

ll c=mod-d\*powmod(b,mod-2)%mod;

while (SZ(C)<SZ(B)+m) C.pb(0);

rep(i,0,SZ(B)) C[i+m]=(C[i+m]+c\*B[i])%mod;

++m;

}

}

return C;

}

int gao(VI a,ll n) {

VI c=BM(a);

c.erase(c.begin());

rep(i,0,SZ(c)) c[i]=(mod-c[i])%mod;

for (int v:c) printf("%d ",v);puts("");

return solve(n,c,VI(a.begin(),a.begin()+SZ(c)));

}

};

int main() {

int k=linear\_seq::gao(VI{7,16,25,50,84,159,277,511,906,1651,2952,5348,9601,17345,31199,56288,101341},10);

for (scanf("%d",&\_);\_;\_--) {

scanf("%d",&n);

printf("%d\n",linear\_seq::gao(VI{0,1,1,2,3,5,8,13,21,34},n-1));

}

}

并查集 整体维护

struct Edge {

int u,v,val;

} edge[maxn];

int head[maxn];

bool cmp(Edge &A,Edge &B){

return A.val<B.val;

};

int fa[maxn];

ULL sum[maxn],cnt[maxn];

inline int getfa(int x){

if (fa[x]==x) return x;

int y=getfa(fa[x]);

if (fa[x]!=y) sum[x]+=sum[fa[x]];

fa[x]=y;

return y;

}

int solve(){

int n,m;

scanf("%d%d",&n,&m);

int i;

REP(i,m)

scanf("%d%d%d",&edge[i].u,&edge[i].v,&edge[i].val);

sort(edge,edge+m,cmp);

FOR(i,1,n) fa[i]=i,sum[i]=0,cnt[i]=1;

REP(i,m){

int x=getfa(edge[i].u),y=getfa(edge[i].v);

if (x==y) continue;

if (cnt[x]>cnt[y]) swap(x,y);

sum[y]+=cnt[x]\*edge[i].val;

sum[x]+=cnt[y]\*edge[i].val;

sum[x]-=sum[y];fa[x]=y;

cnt[y]+=cnt[x];

}ULL ans=0;

FOR(i,1,n){

int x=getfa(i);

ULL val=sum[i];

if (x!=i) val+=sum[x];

ans^=(ULL)i\*val;

}static int x=0;;

printf("Case #%d: %llu\n",++x,ans);

return 0;

}

int main() {

int T;

scanf("%d",&T);

while (T--) solve();

return 0;

}

Trie xor后小于limit最大值

//x xor v->max;

//没注释的是v<limit

//注释的是xor后小于limit

//计数问题有个套路:

//先算出全部,然后for一边容斥

int nxt[maxn\*20\*10][2],tot;

int cnt[maxn\*20\*10];

LL xornum,limit;

void Ins(int &now,int k,int val){

if (!now) now=++tot;

cnt[now]+=val;

if (k==-1) return;

int c=(xornum>>k)&1;

Ins(nxt[now][c],k-1,val);

}

LL Que(int now,int k,bool mark){//mark:have limit

if (!now||!cnt[now]) return -INFF;

if (k==-1) return 0;

int c=(xornum>>k)&1,lim=(limit>>k)&1;

LL ret=-INFF;

if (!lim&&mark){

return (c<<k)+Que(nxt[now][0],k-1,mark);

// return Que(nxt[now][c],k-1,mark);

}else {

ret=(1ll<<k)+Que(nxt[now][c^1],k-1,mark&&!(c&1));

if (ret<0) ret=Que(nxt[now][c],k-1,mark&&(c&1));

// ret=(1ll<<k)+Que(nxt[now][c^1],k-1,mark);

// if (ret<0) ret=Que(nxt[now][c],k-1,0);

}return ret;

}

DP套DP

//题意:麻将胡牌的可能种数

//为了不数漏,方法是这样的:

//首先考虑每个可能情况选择的个数,只可能有3\*3\*2=18种

//然后我们把状态压一下,每种牌型可能的1<<18的状态!

//对这个1<<18的状态进行转移

void print2(int x){

int i;

rREP(i,18) putchar(((x>>i)&1)+'0');

}int encode(int n\_2,int n\_1,int have2){//start from n-2 | n-1

int ret=0;

ret=ret\*3+n\_2;

ret=ret\*3+n\_1;

ret=ret\*2+have2;

return ret;

}void decode(int e,int &n\_2,int &n\_1,int &have2){

have2=e%2;e/=2;

n\_1=e%3;e/=3;

n\_2=e%3;e/=3;

}

void printstatus(int e){

int n\_2,n\_1,have2;

decode(e,n\_2,n\_1,have2);

printf(" %d %d %d ",n\_2,n\_1,have2);

}

int getnextstatus(int status,int k){

int nxtstatus=0,n;

int n\_2,n\_1,have2;

int x\_2,x\_1,xave2;

REP(n,18) if ((status>>n)&1){

decode(n,n\_2,n\_1,have2);

x\_2=n\_1;x\_1=k-n\_2-n\_1;xave2=have2;

if (x\_1>=0){

int x=encode(x\_2,x\_1%3,xave2);

nxtstatus|=(1<<x);

// printstatus(n);printf("->");printstatus(x);printf("(+%d)",k);puts("");

}if (!have2&&x\_1-2>=0){

int x=encode(x\_2,x\_1-2,1);

nxtstatus|=(1<<x);

// printstatus(n);printf("->");printstatus(x);printf("(+%d)",k);puts("");

}

}

// printf("get:%d->%d (k=%d)\n",status,nxtstatus,k);

return nxtstatus;

}

queue<int> Q;

int id[1<<18|7],val[1007];

int tot;

int nxt[1007][7];

void initDP(){

int i,j;tot=0;

int k;//this\_number

Q.push(1);id[0]=++tot;

while (Q.size()){

int status=Q.front();Q.pop();

FOR(k,0,4){//只考虑这里产生2~

int nxtstatus=getnextstatus(status,k);

if (!id[nxtstatus]) id[nxtstatus]=++tot,val[tot]=nxtstatus,Q.push(nxtstatus);

nxt[id[status]][k]=id[nxtstatus];

}

}

// printf("%d\n",tot);

// REP(i,(1<<18)) if (id[i]){

// printf("(%-2d): ",id[i]);

// print2(i);puts("");

// REP(j,18) if ((i>>j)&1) printstatus(j);puts("");

// }

// FOR(i,1,tot){

// printf(" %-2d : ",i);

// print2(val[i]);puts("");

// REP(j,18) if ((val[i]>>j)&1) printstatus(j);puts("");

// }

}

int dp[207][207][78];

inline void update(int &x,int y){

((x+=y)>M)&&(x-=M);

}

int solve(int n,int m){

int i,j,k,t;

FOR(i,0,n+3) FOR(j,0,m) FOR(t,0,68) dp[i][j][t]=0;

dp[0][0][1<<id[encode(0,0,0)]]=1;

FOR(i,0,n+3){

int MAX;

if (i<n) MAX=4;else MAX=0;

FOR(j,0,m){

FOR(t,1,tot) if (dp[i][j][t]){

FOR(k,0,MAX){

int nxtpos=nxt[t][k];

// printf("%d->%d; k=%d\n",t,id[nxtstauts],k);

update(dp[i+1][j+k][nxtpos],dp[i][j][t]);

}

}

}

}int ret=0;

// FOR(t,1,tot) printf("%d: %d\n",t,dp[n+3][m][t]);

FOR(t,1,tot){

if ((val[t]>>encode(0,0,1))&1){

update(ret,dp[n+3][m][t]);

// printf("t=%d\n",t);

}

}

return ret;

}

int main() {

int T;

initDP();

scanf("%d",&T);

while (T--){

int n,m;

static int x=0;

scanf("%d%d",&n,&m);

printf("Case #%d: %d\n",++x,solve(n,m));

}

return 0;

}

虚树

// 题意:问最少去掉几个未标记点可以把所有的标记点全分开

// 做法:建虚树然后树上DP

// 虚树板子,注意:sort过程可以提到外边去

struct Edges {

int to; LL len; int next;

Edges(int \_to=0,LL \_len=0,int \_next=0):to(\_to),len(\_len),next(\_next) {}

} edge[maxn\*2]; int etot;

int head[maxn];

int fa[maxn];

LL uplen[maxn];

int id[maxn],dfn[maxn],idtot;

inline void addedge(int u,int v,LL len) {

edge[++etot]=Edges(v,len,head[u]); head[u]=etot;

}

namespace LCA {//内部和外部dfn不同...

int dep[maxn]; LL len[maxn];

int st\_dfn[maxn],tot;

int ST[maxn\*2][20];//only L

void dfs(int x,int f,int d,LL l) {

int i; dep[x]=d; len[x]=l;

st\_dfn[x]=++tot; ST[tot][0]=x;

::id[++idtot]=x; ::dfn[x]=idtot;

for (i=head[x]; ~i; i=edge[i].next) if (edge[i].to!=f) {

int v=edge[i].to;

::fa[v]=x; ::uplen[v]=edge[i].len;

dfs(v,x,d+1,l+edge[i].len);

ST[++tot][0]=x;

}

}

int t\_t[maxn\*2];

inline void initST(int n) {

int i,j;

FOR(i,1,n\*2) t\_t[i]=t\_t[i>>1]+1;

FOR(i,1,n\*2) {

rep(j,1,t\_t[i]) {

int u=ST[i][j-1],v=ST[i-(1<<(j-1))][j-1];

ST[i][j]=dep[u]<dep[v]?u:v;

}

}

}

inline int lca(int x,int y) {

x=st\_dfn[x]; y=st\_dfn[y];

if (x>y) swap(x,y);

int t=t\_t[y-x+1]-1;

x=ST[x+(1<<t)-1][t]; y=ST[y][t];

return dep[x]<dep[y]?x:y;

}

inline LL dis(int x,int y) {

return len[x]+len[y]-2\*len[lca(x,y)];

}

}

namespace vtree {

int S[maxn],top;

int pid[maxn],mark[maxn];

int vid[maxn],vfa[maxn];

LL vlen[maxn];

int cmp(int x,int y) {

return dfn[x]<dfn[y];

}

void addedge(int u,int v) {

vfa[v]=u; vlen[v]=LCA::dis(u,v);

}

int m;

void vbuild(int n) {

int i; m=0;

sort(pid+1,pid+1+n,cmp);

S[top=1]=pid[1];

mark[pid[1]]=1;

FOR(i,2,n) {

int f=LCA::lca(pid[i-1],pid[i]);

while (top&&LCA::dep[S[top]]>LCA::dep[f]) {

int v; vid[++m]=v=S[top--];

if (top&&LCA::dep[S[top]]>LCA::dep[f]) addedge(S[top],v);

else addedge(f,v);

} if (!top||S[top]!=f) S[++top]=f;

S[++top]=pid[i]; mark[pid[i]]=1;

} while (top-1) addedge(S[top-1],S[top]),vid[++m]=S[top--];

vid[++m]=S[1];

reverse(vid+1,vid+m+1);

}

void vclear() {

int i;

FOR(i,1,m) mark[vid[i]]=0;

}

}

int ans;

int cnt[maxn];

void solve() {

int i;

FOR(i,1,vtree::m) cnt[vtree::vid[i]]=0;

rFOR(i,1,vtree::m) {

int x=vtree::vid[i];

if (vtree::mark[x]) ans+=cnt[x],cnt[x]=1;

else if (cnt[x]>1) ans++,cnt[x]=0;

if (i>1) cnt[vtree::vfa[x]]+=cnt[x];

}

}

int vis[maxn];

int main() {

int i;

int n,q;

scanf("%d",&n);

FOR(i,1,n) head[i]=-1;

FOR(i,1,n-1) {

int u,v;

scanf("%d%d",&u,&v);

addedge(u,v,1); addedge(v,u,1);

} LCA::dfs(1,0,0,0);

LCA::initST(n);

scanf("%d",&q);

while (q--) {

int m,mark=0;

scanf("%d",&m);

FOR(i,1,m) scanf("%d",&vtree::pid[i]);

FOR(i,1,m) vis[vtree::pid[i]]=1;

FOR(i,1,m) if (vis[fa[vtree::pid[i]]]) mark=1;

FOR(i,1,m) vis[vtree::pid[i]]=0;

if (mark) {puts("-1"); continue;}

vtree::vbuild(m);

ans=0;

solve();

vtree::vclear();

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

}

return 0;

}

C(n,n)

LL inv[1000002];//inverse

LL fac[1000002];//Factorial

// 求出的是ax+by=1的解(a,b正负不限,而且挺小的);

// d(gcd)==1时存在逆元;(d!=1)&&(num|d)时,num\*a/d可认为逆元

// (x+p)%p为逆元

// DP:C[i][j]=(C[i-1][j-1]+C[i][j-1])%M

void exgcd(LL a,LL b,LL &d,LL &x,LL &y){

if (!b) {d=a;x=1;y=0;}

else {exgcd(b,a%b,d,y,x);y-=a/b\*x;}

}

// 前面那个线性求逆元的log版2333

int getinv(int n){

if (n==1) return 1;

return (M-M/n)\*(getinv(M%n))%M;

}

LL C(int n,int m){

return fac[n]\*inv[m]%M\*inv[n-m]%M;

}

//Lucas扩展：Kummer定理：

//C(n,k)中的p的幂次的为p进制下n-k借位次数

//e.g.求C(n,0)...C(n,n)的lcm%(1e9+7)

//做法:考虑每个素因子,n转化为p进制后,除了最后的为p-1的都可以借位

//ans=pow(p,k)的乘积

LL lucas(LL n,LL m){//注意MOD不能太大=\_=! Mlogn

return m==0?1:1ll\*C(n%M,m%M)\*lucas(n/M,m/M)%M;

}

int main(){

int i;

fac[0]=1;

FOR(i,1,1000000) fac[i]=i\*fac[i-1]%M;

inv[0]=inv[1]=1;

FOR(i,2,1000000) inv[i]=(M-M/i)\*inv[M%i]%M;

FOR(i,1,1000000) inv[i]=inv[i]\*inv[i-1]%M;// inv(n!)

printf("%I64d",C(10,3));

}

SPLAY\_NEW

int A[maxn];

struct splay\_tree{

struct node{

int val,min,max,add,size,son[2];//add=lazy

bool rev;

void init(int \_val){//开始时T[i].val==a[i-1](线性的);

val=min=max=\_val;size=1;

if (\_val==INF) max=-INF;

add=rev=son[0]=son[1]=0;

}

}T[maxn\*2];//内存池

int fa[maxn\*2],root,tot;

void pushup(int x){

T[x].min=T[x].max=T[x].val;T[x].size=1;

if (T[x].val==INF) T[x].max=-INF;

if (T[x].son[0]){

T[x].min=min(T[x].min,T[T[x].son[0]].min);

T[x].max=max(T[x].max,T[T[x].son[0]].max);

T[x].size+=T[T[x].son[0]].size;

}

if (T[x].son[1]){

T[x].min=min(T[x].min,T[T[x].son[1]].min);

T[x].max=max(T[x].max,T[T[x].son[1]].max);

T[x].size+=T[T[x].son[1]].size;

}

}

void pushdown(int x){

if (x==0) return;

if (T[x].add){

if (T[x].son[0]){

T[T[x].son[0]].val+=T[x].add;

T[T[x].son[0]].min+=T[x].add;

T[T[x].son[0]].max+=T[x].add;

T[T[x].son[0]].add+=T[x].add;

}

if (T[x].son[1]){

T[T[x].son[1]].val+=T[x].add;

T[T[x].son[1]].min+=T[x].add;

T[T[x].son[1]].max+=T[x].add;

T[T[x].son[1]].add+=T[x].add;

}

T[x].add=0;

}

if (T[x].rev){

if (T[x].son[0]) T[T[x].son[0]].rev^=1;

if (T[x].son[1]) T[T[x].son[1]].rev^=1;

swap(T[x].son[0],T[x].son[1]);

T[x].rev=0;

}

}

void rotate(int x,int kind){//zig(1->) zag(0<-)都行

int y=fa[x],z=fa[y];

T[y].son[!kind]=T[x].son[kind],fa[T[x].son[kind]]=y;

T[x].son[kind]=y,fa[y]=x;

T[z].son[T[z].son[1]==y]=x,fa[x]=z;

pushup(y);

}

void splay(int x,int goal){//node x->goal's son

if (x==goal) return;

while (fa[x]!=goal){

int y=fa[x],z=fa[y];

pushdown(z),pushdown(y),pushdown(x);

int rx=T[y].son[0]==x,ry=T[z].son[0]==y;

if (z==goal) rotate(x,rx);

else{

if (rx==ry) rotate(y,ry);

else rotate(x,rx);

rotate(x,ry);

}

}

pushup(x);

if (goal==0) root=x;

}

int select(int pos){//getnode

int u=root;

pushdown(u);

while (T[T[u].son[0]].size!=pos){//这里由于头节点有个-INF 所以不-1

if (pos<T[T[u].son[0]].size) u=T[u].son[0];

else{

pos-=T[T[u].son[0]].size+1;

u=T[u].son[1];

}

pushdown(u);

}

return u;

}

//下面是自己写的一点常用?函数

void update(int l,int r,int val){

int u=select(l-1),v=select(r+1);

splay(u,0);

splay(v,u);

T[T[v].son[0]].min+=val;

T[T[v].son[0]].max+=val;

T[T[v].son[0]].val+=val;

T[T[v].son[0]].add+=val;//lazy

}

void reverse(int l,int r){

int u=select(l-1),v=select(r+1);

splay(u,0);splay(v,u);

T[T[v].son[0]].rev^=1;

}

void revolve(int l,int r,int x){//l~r->循环往后x位

int u=select(r-x),v=select(r+1);

splay(u,0);splay(v,u);

int tmp=T[v].son[0];T[v].son[0]=0;

pushup(v);pushup(u);

u=select(l-1),v=select(l);

splay(u,0);splay(v,u);

fa[tmp]=v;

T[v].son[0]=tmp;

pushup(v);pushup(u);

}

void cut(int l,int r,int x){//l~r->去掉的x位置后 //HDU3487

int u=select(l-1),v=select(r+1);

splay(u,0);splay(v,u);

int tmp=T[v].son[0];

T[v].son[0]=0;

pushup(v);pushup(u);

u=select(x);v=select(x+1);

splay(u,0);splay(v,u);

fa[tmp]=v;

T[v].son[0]=tmp;

pushup(v);pushup(u);

}

int query\_min(int l,int r){

int u=select(l-1),v=select(r+1);

splay(u,0);

splay(v,u);

return T[T[v].son[0]].min;

}

void insert(int x,int val){

int u=select(x),v=select(x+1);

splay(u,0); splay(v,u);

++tot;if (tot==maxn) tot=1;

T[tot].init(val); fa[tot]=v;

T[v].son[0]=tot;

pushup(v);pushup(u);

}

void erase(int x){

int u=select(x-1),v=select(x+1);

splay(u,0);

splay(v,u);

T[v].son[0]=0;

pushup(v);pushup(u);

}

void exchange(int l1,int r1,int l2,int r2){//r1-l1+1?=r2-l2+1 OK

if (l1>l2){swap(l1,l2);swap(r1,r2);}

int u=select(l1-1),v=select(r1+1);

splay(u,0);splay(v,u);

int tmp=T[v].son[0];T[v].son[0]=0;

pushup(v);pushup(u);

l2-=T[tmp].size;r2-=T[tmp].size;

int \_u=select(l2-1),\_v=select(r2+1);

splay(\_u,0);splay(\_v,\_u);

fa[tmp]=\_v;

swap(T[\_v].son[0],tmp);

pushup(\_v);pushup(\_u);

u=select(l1-1),v=select(l1);

splay(u,0);splay(v,u);

fa[tmp]=v;

T[v].son[0]=tmp;

pushup(v);pushup(u);

}

int dfs(int x,int k){//小于k的值个数,会被卡

if (x==0) return 0;

if (T[x].min!=INF&&T[x].min>=k) return 0;

if (T[x].max!=-INF&&T[x].max<k) return T[x].size;

int ret=T[x].val<k;

if (T[x].son[0]) ret+=dfs(T[x].son[0],k);

if (T[x].son[1]) ret+=dfs(T[x].son[1],k);

return ret;

}

int query(int l,int r,int k){//小于k的值个数,会被卡 应该套主席树(但是太长，两个log)

int u=select(l-1),v=select(r+1);

splay(u,0);splay(v,u);

return dfs(T[v].son[0],k);

}

int delbuf[maxn],bufs;

int build(int l,int r){//add\_list

if (l>r) return 0;

++tot;if (tot==maxn) tot=1;

int ret=delbuf[tot];

int mid=(l+r)/2;

T[ret].init(A[mid]);

if (l==r) return ret;

int ls=build(l,mid-1);

int rs=build(mid+1,r);

if (ls) fa[ls]=ret,T[ret].son[0]=ls;

if (rs) fa[rs]=ret,T[ret].son[1]=rs;

pushup(ret);

return ret;

}

void del(int x){

if (x==0) return;

bufs++;if (bufs==maxn) bufs=1;

delbuf[bufs]=x;

del(T[x].son[0]);

del(T[x].son[1]);

}

void Del(int l,int r){

int u=select(l-1),v=select(r+1);

splay(u,0);splay(v,u);

del(T[v].son[0]);

T[v].son[0]=0;

pushup(v);pushup(u);

}

void init(int n){

int i; tot=0;

REP(i,maxn) delbuf[i]=i;

rFOR(i,1,n) A[i+1]=A[i];

A[1]=A[n+2]=-INF;

root=build(1,n+2);

fa[root]=0; T[0].init(-INF);

fa[0]=0;T[0].son[1]=root;T[0].size=0;

}

}T;

积性函数

n=sigma{phi(d)[d|n]} 将phi看作容斥系数

[n=1]=sigma{mu(d)[d|n]} 将i/n化为最简分数

这里可以把gcd或者lcm的式子提出来!

1…n的与n互质数和n\*phi(n)/2

然后，经过推导可能将某些式子化成简单形式就能做了qwq完全不会，智商不够没办法……