**Graph Theory**

**Dijkstra + Heap**

#include <iostream>

#include <stdio.h>

#include <algorithm>

#include <vector>

#include <queue>

#define maxn 10005

#define inf 0x7fffffff

#define pb push\_back

using namespace std;

struct edge{int v,c;}temp1;

struct pnt {int v,c;}temp2;

vector<edge>e[maxn];

priority\_queue<pnt>q;

int n,m,x,y,z,dist[maxn];

bool flag[maxn];

bool operator <(const pnt &a,const pnt &b){return a.c>b.c;}

void dijkstra(){

temp2.v=1,temp2.c=0;

q.push(temp2);

for(int i=2;i<=n;i++)

dist[i]=inf;

while(!q.empty()){

int f=q.top().v;q.pop();

if(flag[f])continue;

flag[f]=true;

for(int i=0,l=e[f].size();i<l;i++)

if(dist[e[f][i].v]>dist[f]+e[f][i].c){

dist[e[f][i].v]=dist[f]+e[f][i].c;

temp2.v=e[f][i].v,temp2.c=dist[e[f][i].v];

q.push(temp2);

}

}

}

**Floyd**

for(int k=1;k<=n;k++)

for(int i=1;i<=n;i++)

for(int j=1;j<=n;j++)

f[i][j]=min(f[i][j],f[i][k]+f[k][j]);

**Hungary**

bool Hungary(int x){

for(int i=1;i<=boy;i++)

if(a[x][i]&&!flag[i]){ flag[i]=true;

if(!dy[i]||Hungary(dy[i])){dx[x]=i,dy[i]=x;return true;}}

return false;

}

for(int i=1;i<=girl;i++)for(int j=1;j<=boy;j++)

if(a[i][j]&&!dx[i]&&!dy[j])ans++,dx[i]=j,dy[j]=i;

for(int i=1;i<=girl;i++)

if(!dx[i]){

memset(flag,0,sizeof(flag));

if(Hungary(i))ans++;

}

**SPFA**

while(!q.empty()){

int f=q.front();q.pop();

flag[f]=false;

for(int i=0,l=e[f].size();i<l;i++)

if(dist[e[f][i].v]>dist[f]+e[f][i].c){

dist[e[f][i].v]=dist[f]+e[f][i].c;

if(!flag[e[f][i].v])

flag[e[f][i].v]=true,q.push(e[f][i].v);

}}

**Dinic**

struct edge{

int v, next;

LL flow;

}e[MAXM];

int first[MAXN], level[MAXN], ecnt;

void addedge(int u, int v, LL flow){

e[++ ecnt].next = first[u], first[u] = ecnt, e[ecnt].v = v, e[ecnt].flow = flow;

e[++ ecnt].next = first[v], first[v] = ecnt, e[ecnt].v = u, e[ecnt].flow = 0;

}

bool make\_level(int S, int T){

memset(level, 0, sizeof(level));

level[S] = 1;

queue <int> Q;

Q.push(S);

while(!Q.empty()){

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

for(int i = first[f]; i != -1; i = e[i].next)

if(e[i].flow && !level[e[i].v]){

level[e[i].v] = level[f] + 1;

Q.push(e[i].v);

if(e[i].v == T) return true;

}

}

return level[T];

}

LL maxflow(int u, int T, LL flow){

if(u == T) return flow;

LL d, temp = 0;

for(int i = first[u]; i != -1 && temp < flow; i = e[i].next)

if(level[e[i].v] == level[u] + 1 && e[i].flow)

if(d = maxflow(e[i].v, T, min(e[i].flow, flow - temp)))

temp += d, e[i].flow -= d, e[i ^ 1].flow += d;

if(!temp) level[u] = 0;

return temp;

}

LL ANS;

void Dinic(int S, int T){

LL d;

while(make\_level(S, T))

while(d = maxflow(S, T, INF))

ANS += d;

}

**Mincost\_maxflow**

struct edge{

int u, v, next, flow, cost;

}e[MAXM];

int ecnt, first[MAXN], dist[MAXN], pre[MAXN];

bool flag[MAXN];

void addedge(int u, int v, int flow, int cost){

//cout << u << ' ' << v << ' ' << flow << ' ' << cost << endl;

e[++ ecnt].next = first[u], first[u] = ecnt, e[ecnt].u = u, e[ecnt].v = v, e[ecnt].flow = flow, e[ecnt].cost = cost;

e[++ ecnt].next = first[v], first[v] = ecnt, e[ecnt].u = v, e[ecnt].v = u, e[ecnt].flow = 0, e[ecnt].cost = -cost;

}

bool make\_map(int S, int T){

for(int i = 0; i < MAXN ; i ++)

dist[i] = INF, pre[i] = -1;

memset(flag, 0, sizeof(flag));

dist[S] = 0;

queue <int> Q;

Q.push(S);

while(!Q.empty()){

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

flag[f] = false;

for(int i = first[f]; i != -1; i = e[i].next)

if(e[i].flow && dist[e[i].v] > dist[f] + e[i].cost){

dist[e[i].v] = dist[f] + e[i].cost,

pre[e[i].v] = i;

if(!flag[e[i].v])

Q.push(e[i].v), flag[e[i].v]=true;

}

}

return dist[T] != INF;

}

int mincost\_maxflow(int S, int T){

int ans = 0;

while(make\_map(S, T)){

int delta = INF, cost = 0;

for(int i = pre[T]; i != -1; i = pre[e[i].u])

delta = min(e[i].flow, delta), cost += e[i].cost;

for(int i = pre[T]; i != -1; i = pre[e[i].u])

e[i].flow -= delta, e[i ^ 1].flow += delta;

ans += delta \* cost;

}

return ans;

}

**Directed graph SCC - Tarjan**

int dfn[MAXN], low[MAXN], scc\_id[MAXN], cnt, idx, scc\_cnt;

vector <int> vec[MAXN];

stack <int> stk;

void dfs(int u){

stk.push(u);

dfn[u] = low[u] = ++ idx;

for(int i = first[u]; i != -1; i = e[i].next){

int v = e[i].v;

if(!dfn[v])

dfs(v), low[u] = min(low[u], low[v]);

else if(!scc\_id[v])

low[u] = min(low[u], dfn[v]);

}

if(dfn[u] == low[u]){

scc\_cnt ++;

while(1){

int x = stk.top(); stk.pop();

vec[scc\_cnt].pb(x);

scc\_id[x] = scc\_cnt;

if(x == u) break;

}

}

}

int n, x;

void gao(){

cnt = idx = 0;

for(int i = 1; i <= n; i ++)

if(!dfn[i]) dfs(i);

}

**Undirected graph – PBC**

void dfs(int x,int f){

low[x]=dfn[x]=++index;

for(int i=0,s=e[x].size();i<s;i++){

int v=e[x][i];

if(!dfn[e[x][i]]){

stack[++cnt][1]=x,stack[cnt][2]=v;

dfs(e[x][i],x);

low[x]=min(low[x],low[e[x][i]]);

if(low[v]>=dfn[x]){

ne[++scnt].clear();

while(1){

int xx=stack[cnt][1],vv=stack[cnt--][2];

if(belong[xx]!=scnt)

ne[scnt].pb(xx),belong[xx]=scnt;

if(belong[vv]!=scnt)

ne[scnt].pb(vv),belong[vv]=scnt;

if(xx==x && vv==v)break;

}

}

}

else if(dfn[v]<dfn[x] && v!=f)

low[x]=min(low[x],dfn[v]);

}

}

**Undirected graph – EBC**

void dfs(int x){

low[x]=dfn[x]=++id;

for(int i=first[x];i!=-1;i=e[i].next)if(!e[i].vis){

int v=e[i].v;

e[i].vis=e[i^1].vis=true;

if(!dfn[v])dfs(v),low[x]=min(low[x],low[v]);

else low[x]=min(low[x],dfn[v]);

}

}

for(int i=1;i<=n;i++)

if(!dfn[i])

dfs(i);

for(int i=0;i<=cnt;i++)

if(low[e[i].v]>dfn[e[i].u])

e[i].cut=e[i^1].cut=true,ans++;

**Bipartite Graph Coloring**

bool dfs2(int x,int X){

for(int i=0,s=e[x].size();i<s;i++)

//if(belong[e[x][i]]==X){

if(col[e[x][i]]==col[x])return false;

if(!col[e[x][i]]){

col[e[x][i]]=3-col[x];

if(!dfs2(e[x][i],X))return false;

}

//}

return true;

}

**Doubling - LCA (NlgN)**

void dfs(int x,int pre,int depth,int dis){

dist[x]=dis;deep[x]=depth;

for(int i=first[x];i;i=e[i].next)if(e[i].v!=pre)

dfs(e[i].v,x,depth+1,dis+e[i].cost),father[e[i].v][0]=x;

}

void init(){

for(int i=1;(1<<i)<=n;i++)

for(int j=1;j<=n;j++)

father[j][i]=father[father[j][i-1]][i-1];

}

int lca(int x,int y){

if(deep[x]<deep[y])swap(x,y);

int temp=deep[x]-deep[y];

for(int i=0;(1<<i)<=temp;i++)

if(temp&(1<<i))x=father[x][i];

if(x==y)return x;

for(int i=20;i>=0;i--)

if(father[x][i]!=father[y][i])x=father[x][i],y=father[y][i];

return father[x][0];

}

**Divide and Conquer on Tree (Point) – Zju4812**

int first[MAXN], ecnt, size[MAXN];

int heavy, totsize, mx, num[MAXN];

bool del[MAXN];

struct edge{

int next, v;

} e[MAXN << 1];

void addedge(int u, int v){

e[++ ecnt].next = first[u], first[u] = ecnt, e[ecnt].v = v;

}

void dfs\_size(int u, int pre){

size[u] = 1;

for(int i = first[u]; i != -1; i = e[i].next)

if(e[i].v != pre && !del[e[i].v])

dfs\_size(e[i].v, u), size[u] += size[e[i].v];

}

void dp(int u, int pre){

int tmp = 0;

for(int i = first[u]; i != -1; i = e[i].next)

if(e[i].v != pre && !del[e[i].v])

dp(e[i].v, u), tmp = max(tmp, size[e[i].v]);

tmp = max(tmp, totsize - size[u]);

if(tmp < mx) mx = tmp, heavy = u;

}

void dfs\_get\_heavy(int u){

heavy = -1, mx = INF;

dfs\_size(u, 0), totsize = size[u];

dp(u, 0);

}

pair<int, LL> flag[mod];

vector<pair<int, int> > vec;

LL ID;

pair<int, int> ANS;

void go(int x, int y){

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

ANS = min(ANS, mp(x, y));

}

int m;

void gao(int u, int pre, int tot, int k){

tot = (LL) tot \* num[u] % mod;

for(int i = first[u]; i != -1; i = e[i].next){

int v = e[i].v;

if(del[e[i].v] || e[i].v == pre) continue;

gao(v, u, tot, k);

}

k = (LL) m \* inv[(LL) tot \* k % mod] % mod;

if(flag[k].second == ID)

go(flag[k].first, u);

vec.pb(mp(tot, u));

}

void dfs(int u){

dfs\_get\_heavy(u);

u = heavy;

del[u] = true;

if(totsize == 1) return;

ID ++;

flag[1] = mp(u, ID);

for(int i = first[u]; i != -1; i = e[i].next)

if(!del[e[i].v]){

vec.clear();

gao(e[i].v, 0, 1, num[u]);

for(int j = 0; j < vec.size(); j ++){

if(flag[vec[j].first].second == ID)

flag[vec[j].first].first = min(flag[vec[j].first].first, vec[j].second);

else flag[vec[j].first].first = vec[j].second, flag[vec[j].first].second = ID;

}

}

for(int i = first[u]; i != -1; i = e[i].next)

if(!del[e[i].v])

dfs(e[i].v);

}

**Data Structures**

**Trie\_Map**

void insert(char \*word){

int p = 0;

for(int i = 0, l = strlen(word); i < l; i ++){

int tmp = word[i] - 'a' + 1;

if(!tree[p].son[tmp])

tree[p].son[tmp] = ++ sz;

p = tree[p].son[tmp];

}

tree[p].mk = true;

}

void Build\_Trie\_Map(){

queue <int> Q;

for(int i = 1; i <= m; i ++)

if(tree[0].son[i])

Q.push(tree[0].son[i]);

while(!Q.empty()){

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

for(int i = 1; i <= m; i ++){

if(tree[f].son[i]){

tree[tree[f].son[i]].fail = tree[tree[f].fail].son[i];

if(tree[tree[tree[f].fail].son[i]].mk)

tree[tree[f].son[i]].mk = true;

Q.push(tree[f].son[i]);

}

else

tree[f].son[i] = tree[tree[f].fail].son[i];

}

}

}

**01\_trie**

void insert(ll x){

int p=0;//注意建树的顺序..应该从高位往低位建..

for(int i=50;i>=0;i--){

bool temp=x&(1LL<<i);

if(!tree[p].son[temp])

tree[p].son[temp]=++size;

p=tree[p].son[temp];

}

}

ll check(ll x){

ll ans=0;

int p=0;

for(int i=50;i>=0;i--){

bool temp=!(x&((1LL)<<i));

if(tree[p].son[temp])

p=tree[p].son[temp],ans|=(1LL<<i);

else

p=tree[p].son[!temp];

}

return ans;

}

**Suffix Array**

int wa[MAXN], wb[MAXN], S[MAXN];

int sa[MAXN], rank[MAXN], height[MAXN];

void da(char \*ch, int n, int m){

int i, j, p = 0, \*X = wa, \*Y = wb;

for(i = 0; i < m; i ++) S[i] = 0;

for(i = 0; i < n; i ++) S[X[i] = ch[i]] ++;

for(i = 1; i < m; i ++) S[i] += S[i - 1];

for(i = n - 1; i >= 0; i --) sa[-- S[X[i]]] = i;

for(j = 1; p < n; j \*= 2, m = p){

for(p = 0, i = n - j; i < n; i ++) Y[p ++] = i;

for(i = 0; i < n; i ++) if(sa[i] >= j) Y[p ++] = sa[i] - j;

for(i = 0; i < m; i ++) S[i] = 0;

for(i = 0; i < n; i ++) S[X[Y[i]]] ++;

for(i = 1; i < m; i ++) S[i] += S[i - 1];

for(i = n - 1; i >= 0; i --) sa[-- S[X[Y[i]]]] = Y[i];

swap(X, Y);

for(p = 1, X[sa[0]] = 0, i = 1; i < n; i ++)

X[sa[i]] = (Y[sa[i-1]] == Y[sa[i]] && Y[sa[i - 1] + j] == Y[sa[i] + j]) ? p - 1 : p ++;

}

}

void calcheight(char \*ch, int n){

int i, j, k = 0;

for(i = 0; i <= n; i ++) rank[sa[i]] = i;

for(i = 0; i < n; height[rank[i ++]] = k)

for(k ? k -- : 0, j = sa[rank[i] - 1]; ch[i + k] == ch[j + k]; k ++);

}

int len1, lensum;

char ch[MAXN];

int main(){

//freopen("in.txt", "r", stdin);

scanf(" %s", ch);

len1 = strlen(ch);

ch[len1] = '$';

scanf(" %s", ch + len1 + 1);

lensum = strlen(ch);

ch[lensum] = '\0';

da(ch, lensum + 1, 200), calcheight(ch, lensum);

/\*cout << ch << endl;

for(int i = 0; i <= lensum; i ++)

cout << sa[i] << ' ' << rank[i] << ' ' << height[i] << endl;\*/

int ans = 100000;

for(int i = 1; i <= lensum; i ++){

if(height[i] > 0){

if((sa[i] > len1 && sa[i - 1] < len1) || (sa[i] < len1 && sa[i - 1] > len1)){

if(height[i - 1] >= height[i] || height[i + 1] >= height[i]) continue;

int k = max(height[i - 1], height[i + 1]);

ans = min(ans, k + 1);

}

}

}

if(ans == 100000) cout << -1 << endl;

else cout << ans << endl;

return 0;

}}

**Multi segment tree**

#define lc(x) (x<<1)

#define rc(x) (lc(x)+1)

struct segtree{int l,r,s,mark;int mid(){return (l+r)>>1;}}tree[16\*maxn];

void buildtree(int l,int r,int xx,int delta){

int x=xx+delta;

tree[x].l=l,tree[x].r=r;

if(l==r)return;

int mid=tree[x].mid();

buildtree(l,mid,lc(xx),delta);

buildtree(mid+1,r,rc(xx),delta);//注意这里是rc(xx)..

}

void add\_mark(int l,int r,int ad,int xx,int delta){

int x=xx+delta;

if(tree[x].l>=l&&tree[x].r<=r){

tree[x].mark+=ad;return;

}

if(tree[x].mark){//pushdown tree[delta+lc(xx)].mark+=tree[x].mark,tree[delta+rc(xx)].mark+=tree[x].mark;

tree[x].mark=0;

}

int mid=tree[x].mid();

if(l<=mid)add\_mark(l,r,ad,lc(xx),delta);

if(r>mid)add\_mark(l,r,ad,rc(xx),delta);

//update();

}

int query(int pnt,int xx,int delta){

int x=xx+delta;

if(tree[x].l==tree[x].r){return tree[x].mark;}

if(tree[x].mark){//pushdown

tree[delta+lc(xx)].mark+=tree[x].mark,tree[delta+rc(xx)].mark+=tree[x].mark;

tree[x].mark=0;

}

int mid=tree[x].mid();

if(pnt<=mid)return query(pnt,lc(xx),delta);

else return query(pnt,rc(xx),delta);

}

**Descartes tree ( Big on top )**

tree[root=0].val=0x7fffffff;

void insert(int x){

int p=x-1;//因为x-1一定在最右的路径的末端(没有右儿子)

while(tree[p].val<tree[x].val)p=tree[p].f;

if(!p)root=x;

tree[x].l=tree[p].r,tree[tree[p].r].f=x;

tree[x].f=p,tree[p].r=x;

}

**Splay Tree**

struct splay{

int l, r, f, sz, num[2], gcd[2];

void init(){

l = r = f = sz = num[0] = num[1] = gcd[0] = gcd[1] = 0;

}

}tree[MAXN];

int root, sz;

void debug(){

cout << "Root : " << root << endl;

for(int i = 0; i <= sz; i ++){

cout << i << ' ' << tree[i].l << ' ' << tree[i].r << ' ' << tree[i].sz << ' ' << tree[i].num[0] << ' ' << tree[i].num[1] << ' ' << tree[i].gcd[0] << ' ' << tree[i].gcd[1] << endl;

}

}

int gcd(int x, int y){

return !y ? x : gcd(y, x % y);

}

void update(int x){

tree[x].gcd[0] = gcd(gcd(tree[x].num[0], tree[tree[x].l].gcd[0]), tree[tree[x].r].gcd[0]);

tree[x].gcd[1] = gcd(gcd(tree[x].num[1], tree[tree[x].l].gcd[1]), tree[tree[x].r].gcd[1]);

tree[x].sz = 1 + tree[tree[x].l].sz + tree[tree[x].r].sz;

}

void zig(int x){

int y = tree[x].f, z = tree[y].f, A = tree[x].r;

if(tree[z].l == y) tree[z].l=x;

else tree[z].r = x;

tree[x].f = z, tree[y].f = x;

tree[y].l = A, tree[x].r = y;

if(A) tree[A].f = y;

update(y);

}

void zag(int x){

int y = tree[x].f, z = tree[y].f, A = tree[x].l;

if(tree[z].l == y) tree[z].l = x;

else tree[z].r = x;

tree[x].f = z, tree[y].f = x;

tree[y].r = A, tree[x].l = y;

if(A) tree[A].f = y;

update(y);

}

void splay(int x){

int y = tree[x].f, z = tree[y].f;

while(y){

if(!z){

if(x == tree[y].l) zig(x);

else zag(x);

break;

}

if(x == tree[y].l && y == tree[z].l) zig(y), zig(x);

else if(x == tree[y].l && y == tree[z].r) zig(x), zag(x);

else if(x == tree[y].r && y == tree[z].r) zag(y), zag(x);

else zag(x), zig(x);

y = tree[x].f, z = tree[y].f;

}

update(x);

root = x;

}

int find(int M){

int x = root;

while(1){

if(tree[tree[x].l].sz == M - 1) break;

if(tree[tree[x].l].sz < M){

M -= (tree[tree[x].l].sz + 1), x = tree[x].r;

}

else x = tree[x].l;

}

return x;

}

void insert(int x, int num, int sta){

x = find(x), splay(x);

sz ++;

tree[sz].init();

tree[sz].sz = 1, tree[sz].gcd[sta] = tree[sz].num[sta] = num;

if(tree[x].r == 0){

tree[sz].f = x, tree[x].r = sz;

}

else{

x = tree[x].r;

while(tree[x].l) x = tree[x].l;

tree[sz].f = x, tree[x].l = sz;

}

splay(sz);

}

void erase(int x){

x = find(x), splay(x);

if(!tree[x].l){

tree[tree[x].r].f = 0;

root = tree[x].r;

}

else{

int y = tree[x].l;

tree[y].f = 0;

while(tree[y].r) y = tree[y].r;

splay(y);

tree[y].r = tree[x].r;

if(tree[x].r) tree[tree[x].r].f = y;

update(y);

}

tree[x].init();

}

void gao(int x, int num){

x = find(x), splay(x);

if(num == -1){

tree[x].gcd[0] = gcd(gcd(tree[x].num[1], tree[tree[x].l].gcd[0]), tree[tree[x].r].gcd[0]);

tree[x].gcd[1] = gcd(gcd(tree[x].num[0], tree[tree[x].l].gcd[1]), tree[tree[x].r].gcd[1]);

swap(tree[x].num[1], tree[x].num[0]);

}

else{

int sta;

if(tree[x].num[0] != 0) sta = 0; else sta = 1;

tree[x].num[sta] = num;

tree[x].gcd[sta] = gcd(gcd(tree[x].num[sta], tree[tree[x].l].gcd[sta]), tree[tree[x].r].gcd[sta]);

}

}

int query(int x0, int x1, int sta){

x0 = find(x0), splay(x0);

tree[tree[x0].r].f = 0;

x1 = find(x1), splay(x1);

root = x0;

tree[x1].f = x0, tree[x0].r = x1;

update(x0);

return tree[tree[x1].l].gcd[sta];

}

int n, num[MAXN], sta[MAXN];

int build(int l, int r, int f){

if(l > r) return 0;

int x = l + r >> 1;

tree[x].init();

if(f == 0) root = x;

tree[x].f = f;

if(x >= 2 && x <= n + 1)

tree[x].num[sta[x - 1]] = tree[x].gcd[sta[x - 1]] = num[x - 1];

tree[x].l = build(l, x - 1, x);

tree[x].r = build(x + 1, r, x);

update(x);

return x;

}

void init(){

for(int i = 1; i <= n; i ++)

scanf("%d%d", &num[i], &sta[i]);

build(1, n + 2, 0);

sz = n + 2;

//debug();

}

**Splay\_in\_segment-tree**

//maintain(北京集训队)

#include <iostream>

#include <math.h>

#include <stdio.h>

#define maxn 1000005

#define lc(x) (x<<1)

#define rc(x) (lc(x)+1)

using namespace std;

struct seg\_tree{int l,r;int mid(){return l+r>>1;}}seg[4\*maxn];

struct splay{int l,r,f,size,key;}tree[4\*maxn];

int n,m,color[maxn],prev[maxn],temp[maxn],root[4\*maxn],tree\_size;

/\*void pushdown(int x){

//平衡树的标记下传和线段树一样,在给一个节点表上标记后同时对这个节点进行值的修改

//理论上说每次操作都要pushdown的,但如果每次操作之前都先进行提根的splay操作的话就不用刻意pushdown了..因此pushdown可以只在splay和update函数中用到

if(tree[x].rev){

if(tree[x].l){

tree[tree[x].l].rev ^= 1;

swap(tree[tree[x].l].l,tree[tree[x].l].r);

swap(tree[tree[x].l].lmax,tree[tree[x].l].rmax);

}

if(tree[x].r){

tree[tree[x].r].rev ^= 1;

swap(tree[tree[x].r].l,tree[tree[x].r].r);

swap(tree[tree[x].r].lmax,tree[tree[x].r].rmax);

}

tree[x].rev = 0;

}

if(tree[x].mark){

if(tree[x].l){

tree[tree[x].l].key = tree[x].key, tree[tree[x].l].mark = true;

tree[tree[x].l].sum = tree[tree[x].l].size \* tree[tree[x].l].key;

tree[tree[x].l].max = tree[tree[x].l].lmax = tree[tree[x].l].rmax = tree[tree[x].l].key < 0 ? tree[tree[x].l].key: tree[tree[x].l].sum;

}

if(tree[x].r){

tree[tree[x].r].key = tree[x].key, tree[tree[x].r].mark = true;

tree[tree[x].r].sum = tree[tree[x].r].size \* tree[tree[x].r].key;

tree[tree[x].r].max = tree[tree[x].r].lmax = tree[tree[x].r].rmax = tree[tree[x].r].key < 0 ? tree[tree[x].r].key: tree[tree[x].r].sum;

}

tree[x].mark=0;

}

}\*/

void update(int x){

pushdown(x);

tree[x].size=tree[tree[x].l].size+tree[tree[x].r].size+1;

}

void zig(int x){

int y=tree[x].f,z=tree[y].f,A=tree[x].r;

pushdown(y),pushdown(x);

if(tree[z].l==y)tree[z].l=x;

else tree[z].r=x;

tree[x].f=z,tree[y].f=x;

tree[y].l=A,tree[x].r=y;

if(A)tree[A].f=y;

update(y);

}

void zag(int x){

int y=tree[x].f,z=tree[y].f,A=tree[x].l;

pushdown(y),pushdown(x);

if(tree[z].l==y)tree[z].l=x;

else tree[z].r=x;

tree[x].f=z,tree[y].f=x;

tree[y].r=A,tree[x].l=y;

if(A)tree[A].f=y;

update(y);

}

void splay(int x){

int y=tree[x].f,z=tree[y].f;

//这里不用root=x,每次进行splay后最后再改变root..

while(y){

pushdown(x);

if(!z){

if(x==tree[y].l)zig(x);

else zag(x);

break;

}

if(x==tree[y].l&&y==tree[z].l)

zig(y),zig(x);

else if(x==tree[y].l&&y==tree[z].r)

zig(x),zag(x);

else if(x==tree[y].r&&y==tree[z].r)

zag(y),zag(x);

else

zag(x),zig(x);

y=tree[x].f,z=tree[y].f;

}update(x);

}

void insert(int &x,int key,int father){

//lgn建立节点的正确姿势

//如果是利用splay对一条链的性质的话则这个insert完全用不上..

if(!x){

tree\_size++;

tree[tree\_size].f=father,tree[tree\_size].key=key,tree[tree\_size].size=1;

x=tree\_size;

return;

}

if(key<=tree[x].key)insert(tree[x].l,key,x);

else insert(tree[x].r,key,x);

update(x);

}

int find(int x,int key){

//找小于key的节点个数(小于等于key-1的节点个数)

int temp=0;key--;

while(x){

if(key>=tree[x].key)temp+=tree[tree[x].l].size+1,x=tree[x].r;

else x=tree[x].l;

}

return temp;

}

int Find(int x,int M){

//找排名M的数

while(1){

if(tree[tree[x].l].size == M-1)break;

if(tree[tree[x].l].size < M)

x = tree[x].r , M -= (tree[tree[x].l].size+1);//这里要注意多减一个1

else

x = tree[x].l;

}

return x;

}

void erase(int &X,int key){

//删除一个节点值为key的节点

int x=X;

while(1){

if(key==tree[x].key)break;

if(key<tree[x].key)x=tree[x].l;

else x=tree[x].r;

}

splay(x);

if(!tree[x].l){

tree[tree[x].r].f=0;

X=tree[x].r;return;

}

int p=tree[x].l;tree[p].f=0;

while(tree[p].r)p=tree[p].r;

splay(p);

//把它左子树的最后节点提上来

tree[p].r=tree[x].r;

if(tree[x].r)tree[tree[x].r].f=p;

update(p);

X=p;

}

void buildtree(int l,int r,int x){

seg[x].l=l,seg[x].r=r;

//splay之后改变root

for(int i=l;i<=r;i++)

insert(root[x],prev[i],0),splay(tree\_size),root[x]=tree\_size;

if(l==r)return;

int mid=seg[x].mid();

buildtree(l,mid,lc(x));

buildtree(mid+1,r,rc(x));

}

int query(int l,int r,int x){

if(l<=seg[x].l&&seg[x].r<=r)

return find(root[x],l);

int mid=seg[x].mid();

if(r<=mid)return query(l,r,lc(x));

else if(l>mid)return query(l,r,rc(x));

else return query(l,r,lc(x))+query(l,r,rc(x));

}

void change(int pre,int now,int pos,int x){

erase(root[x],pre);

insert(root[x],now,0);

if(seg[x].l==seg[x].r)return;

int mid=seg[x].mid();

if(pos<=mid)change(pre,now,pos,lc(x));

else change(pre,now,pos,rc(x));

}

void cchange(int x,int col){

int tmp;

prev[x]=0;

for(int i=1;i<=n;i++){

if(prev[i]==x){

change(x,prev[x],i,1);

prev[i]=prev[prev[x]];

}

if(i<x&&color[i]==col)prev[x]=i;

if(i>x&&prev[i]<x&&color[i]==col){

change(prev[i],x,i,1);

prev[i]=x;

}

}

color[x]=col;

}

int main(){

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

for(int i=1;i<=n;i++)

scanf("%d",&color[i]);

for(int i=1;i<=n;i++)

prev[i]=temp[color[i]],temp[color[i]]=i;

buildtree(1,n,1);

char A;int x,y;

while(m--){

scanf(" %c%d%d",&A,&x,&y);

if(A=='Q')printf("%d\n",query(x,y,1));

else{

int p=prev[x];

cchange(x,y),change(p,prev[x],x,1);

}

}

return 0;

}

**2D Segment Tree**

struct \_2dseg{

int mn, mx;

} tree[MAXN << 2][MAXN << 2];

int n;

pair<int, int> calcy(int x, int y, int ll, int rr, int ly, int ry){

if(ly <= ll && rr <= ry)

return mp(tree[x][y].mn, tree[x][y].mx);

int mid = ll + rr >> 1;

int mx = -INF, mn = INF;

if(ly <= mid){

pair<int, int> tmp = calcy(x, lc(y), ll, mid, ly, ry);

mn = min(mn, tmp.first);

mx = max(mx, tmp.second);

}

if(mid < ry){

pair<int, int> tmp = calcy(x, rc(y), mid + 1, rr, ly, ry);

mn = min(mn, tmp.first);

mx = max(mx, tmp.second);

}

return mp(mn, mx);

}

pair<int, int> calcx(int x, int ll, int rr, int lx, int rx, int ly, int ry){

if(lx <= ll && rr <= rx)

return calcy(x, 1, 1, n, ly, ry);

int mid = ll + rr >> 1;

int mx = -INF, mn = INF;

if(lx <= mid){

pair<int, int> tmp = calcx(lc(x), ll, mid, lx, rx, ly, ry);

mn = min(mn, tmp.first);

mx = max(mx, tmp.second);

}

if(mid < rx){

pair<int, int> tmp = calcx(rc(x), mid + 1, rr, lx, rx, ly, ry);

mn = min(mn, tmp.first);

mx = max(mx, tmp.second);

}

return mp(mn, mx);

}

void gaoy(int x, int y, int ll, int rr, int posy, int num, bool leaf){

if(ll == rr){

if(leaf) tree[x][y].mn = tree[x][y].mx = num;

else{

tree[x][y].mn = min(tree[lc(x)][y].mn, tree[rc(x)][y].mn);

tree[x][y].mx = max(tree[lc(x)][y].mx, tree[rc(x)][y].mx);

}

return;

}

int mid = ll + rr >> 1;

if(posy <= mid) gaoy(x, lc(y), ll, mid, posy, num, leaf);

else gaoy(x, rc(y), mid + 1, rr, posy, num, leaf);

tree[x][y].mn = min(tree[x][lc(y)].mn, tree[x][rc(y)].mn);

tree[x][y].mx = max(tree[x][lc(y)].mx, tree[x][rc(y)].mx);

}

void gaox(int x, int ll, int rr, int posx, int posy, int num){

if(ll == rr){

gaoy(x, 1, 1, n, posy, num, true);

return;

}

int mid = ll + rr >> 1;

if(posx <= mid) gaox(lc(x), ll, mid, posx, posy, num);

else gaox(rc(x), mid + 1, rr, posx, posy, num);

gaoy(x, 1, 1, n, posy, num, false);

}

**Dynamic Programming**

**Sparse Table**

#include <iostream>

#include <stdio.h>

#define maxn 50010

using namespace std;

int smin[maxn][21],smax[maxn][21],a[maxn],log[maxn];

int main()

{

int q,x,y,n,temp;

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

for(int i=1;i<=n;i++){

scanf("%d",&a[i]);

smin[i][0]=a[i];

smax[i][0]=a[i];

}

for(int i=2;i<=n;i++)

log[i]=log[i/2]+1;

for(int i=1;i<=log[n];i++)

for(int j=1;j+(1<<(i-1))<=n;j++)

smin[j][i]=min(smin[j][i-1],smin[j+(1<<(i-1))][i-1]),

smax[j][i]=max(smax[j][i-1],smax[j+(1<<(i-1))][i-1]);

while(q--){

scanf("%d%d",&x,&y);

temp=log[y-x+1];

printf("%d\n",max(smax[x][temp],smax[y-(1<<temp)+1][temp])-min(smin[x][temp],smin[y-(1<<temp)+1][temp]));

}

system("pause");

return 0;

}

**Digit DP**

#include <iostream>

#include <stdio.h>

#include <string.h>

#define ll long long

#define mod 1000000007

using namespace std;

ll n,m,f[20][2][7][7],f2[20][2][7][7],f3[20][2][7][7],ten[20],dig[20];

int t;

ll mul(ll a,ll b){return (a\*b)%mod;}

ll dp(int pos,bool \_7,int dsum,int sum,bool flag){

if(pos==-1)return (!\_7)&&(dsum!=0)&&(sum!=0);

if(\_7)return 0;

if(!flag&&f[pos][\_7][dsum][sum]!=-1)return f[pos][\_7][dsum][sum];

ll res=0;

for(int i=0,l=flag?dig[pos]:9;i<=l;i++)

res=(res+dp(pos-1,\_7|i==7,(dsum+i)%7,(sum\*10+i)%7,flag&&i==l))%mod;

return flag?res:f[pos][\_7][dsum][sum]=res;

}

ll dp2(int pos,bool \_7,int dsum,int sum,bool flag){

if(pos==-1)return 0;

if(!flag&&f2[pos][\_7][dsum][sum]!=-1)return f2[pos][\_7][dsum][sum];

ll res=0;

for(int i=0,l=flag?dig[pos]:9;i<=l;i++){

ll has=dp(pos-1,\_7|i==7,(dsum+i)%7,(sum\*10+i)%7,flag&&i==l);

ll tmp=mul(i,ten[pos]);

tmp=mul(tmp,has);

res=(res+dp2(pos-1,\_7|i==7,(dsum+i)%7,(sum\*10+i)%7,flag&&i==l))%mod;

res=(res+tmp)%mod;

}

return flag?res:f2[pos][\_7][dsum][sum]=res;

}

ll dp3(int pos,bool \_7,int dsum,int sum,bool flag){

if(pos==-1)return 0;

if(!flag&&f3[pos][\_7][dsum][sum]!=-1)return f3[pos][\_7][dsum][sum];

ll res=0;

for(int i=0,l=flag?dig[pos]:9;i<=l;i++){

ll k1=dp(pos-1,\_7|i==7,(dsum+i)%7,(sum\*10+i)%7,flag&&i==l);

ll k2=dp2(pos-1,\_7|i==7,(dsum+i)%7,(sum\*10+i)%7,flag&&i==l);

ll tmp=mul(i,ten[pos]);

res=(res+dp3(pos-1,\_7|i==7,(dsum+i)%7,(sum\*10+i)%7,flag&&i==l))%mod;

res=(res+mul(2,mul(tmp,k2)))%mod;

res=(res+mul(k1,mul(tmp,tmp)))%mod;

}

return flag?res:f3[pos][\_7][dsum][sum]=res;

}

ll go(ll x){int cnt=0;

while(x)dig[cnt++]=x%10,x/=10;

return dp3(cnt-1,0,0,0,1)%mod;

}

int main(){

scanf("%d",&t);

memset(f,-1,sizeof(f));

memset(f2,-1,sizeof(f2));

memset(f3,-1,sizeof(f3));

ten[0]=1;

for(int i=1;i<20;i++)ten[i]=(ten[i-1]\*10)%mod;

while(t--){

scanf("%I64d%I64d",&n,&m);

printf("%I64d\n",((go(m)-go(n-1))%mod+mod)%mod);

}

system("pause");

return 0;

}

**Mathematics**

**Matrix**

**Determinant\_mod ( n^3lgn )**

ll calc\_det(ll a[][201],int n){

ll ret=1;

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

for(int j=i+1;j<n;j++)

while(a[j][i]){

ll t=a[i][i]/a[j][i];

for(int k=i;k<n;k++)

a[i][k]=(a[i][k]-a[j][k]\*t)%mod;

for(int k=i;k<n;k++)

swap(a[i][k],a[j][k]);

ret=-ret;

}

if(a[i][i]==0)return 0;

ret=ret\*a[i][i],ret%=mod;

}

ret=((ret%mod)+mod)%mod;

return ret;

}

**Matrix Multiplication ( size^3lgn )**

struct Matrix{

LL base[88][88];

void init(){

memset(base, 0, sizeof(base));

}

void init2(){

init();

for(int i = 0; i < size; i ++) base[i][i]=1;

}

void debug(){

for(int i = 0; i < size; i ++){

for(int j = 0; j < size; j ++)

cout << base[i][j] << ' ';

cout << endl;

}

cout << endl;

}

}start, trans, temp, ans;

void copy(Matrix &A, Matrix B){

for(int i = 0; i < size; i ++)

for(int j = 0; j < size; j ++)

A.base[i][j] = B.base[i][j];

}

void go(Matrix &A, Matrix B){

Matrix temp;

temp.init();

for(int i = 0; i < size; i ++)

for(int j = 0; j < size; j ++)

for(int k = 0; k < size; k ++)

temp.base[i][j] += A.base[i][k] \* B.base[k][j],

temp.base[i][j] %= mod;

copy(A, temp);

}

void Mul(LL x){

while(x){

if(x & 1) go(start, trans);

go(trans, trans);

x >>= 1;

}

}

**Gauss Elimination**

**Gauss Elimination – Module ( n^3lgn )**

void gauss(int n,int m){

//n和m为系数矩阵的大小

int i,j,k;

for(i=0,j=0;i<n&&j<m;j++){

for(k=i;k<n;k++)if(a[k][j])break;

if(k<n){

for(int r=0;r<=m;r++)swap(a[i][r],a[k][r]);

//这里从第i+1行开始消就可以了..只要保证是行阶梯就可以

for(int r=i+1;r<n;r++)

if(a[r][j]){

int b1=a[i][j],b2=a[r][j];

for(int t=0;t<=m;t++)

a[r][t]=((b1\*a[r][t]-b2\*a[i][t])%7+7)%7;

}

i++;

}

}//判断无解一定要在最后..

for(int r=i;r<n;r++)if(a[r][m]!=0){

printf("Inconsistent data.\n");

return;

}

if(i<m){

printf("Multiple solutions.\n");

return;

}

//从m-1开始是因为如果一定有解的话一定是一个严格的倒三角形式..

for(int r=m-1;r>=0;r--){

int tmp=0;

for(int t=r+1;t<m;t++)

tmp+=a[r][t]\*ans[t],tmp%=7;

while((a[r][m]-tmp)%a[r][r])a[r][m]+=7;

//这里能够这么做的原因是7是一个质数..因此保证了这么加一定可以在6步之内加出一个解来..

ans[r]=(a[r][m]-tmp)/a[r][r],ans[r]=(ans[r]%7+7)%7;

}

for(int i=0;i<m;i++){

if(ans[i]<3)ans[i]+=7;

printf("%d%s",ans[i],i==m-1?"\n":" ");

}

}

**Gauss Elimination – Real number ( n^3 )**

void gauss(int N,int M){

int i,j,k;

for(i=0,j=0;i<N&&j<M;j++){

k=i;

for(int r=i+1;r<N;r++)

if(fabs(a[k][j])<fabs(a[r][j]))

k=r;

//找到一个大一点的精度好控制一些..但是这种写法是基于一定有解的

if(k<N){

if(i!=k)for(int t=0;t<=M;t++)swap(a[i][t],a[k][t]);

//如果这里是从0开始枚举的话很容易就精度溢出了..因为每次都得消一遍..

for(int r=i+1;r<N;r++)

if(fabs(a[r][j])>eps){//这里一定要用eps来判断..

double tt=a[i][j]/a[r][j];

//注意这里tt的顺序如果是a[r][j]/a[i][j]的话就很可能精度不够..不过是为啥..?..

for(int t=M;t>=0;t--)

a[r][t]-=a[i][t]/tt;

}

i++;

}

}

for(int r=M-1;r>=0;r--){

if(a[r][r]==0.0)continue;//有必要..因为某些点本身不会被涉及到..

double tmp=0.0;

for(int t=r+1;t<M;t++)

tmp+=a[r][t]\*ans[t];

ans[r]=(a[r][M]-tmp)/a[r][r];

}

}

**Gauss Elimination – Find Free Variables (??)**

void dfs(int r,int co,int tot){

if(tot>ANS)return;

if(co==-1){

ANS=min(ANS,tot);return;

}

int first;

for(int i=n-1;i>=0;i--)if(a[r][i])first=i;

if(co==first){

ans[first]=a[r][n];

for(int j=first+1;j<n;j++)

ans[first]^=(ans[j]\*a[r][j]);

dfs(r-1,first-1,tot+ans[first]);

}

else{

ans[co]=1;dfs(r,co-1,tot+1);

ans[co]=0;dfs(r,co-1,tot);

}

}

void gauss(int n,int m){

//n和m为系数矩阵的大小

int i,j,k;

for(i=0,j=0;i<n&&j<m;j++){

k=i;

for(;k<n;k++)if(a[k][j])break;

if(k<n){

for(int r=0;r<=m;r++)swap(a[i][r],a[k][r]);

//这里从第i+1行开始消就可以了..只要保证是行阶梯就可以

for(int r=i+1;r<n;r++)

if(a[r][j])

for(int t=0;t<=m;t++)

a[r][t]^=a[i][t];

i++;

}

}//判断无解一定要在最后..

for(int r=i;r<n;r++)if(a[r][m]!=0){

printf("impossible\n");

return;

}

dfs(i-1,m-1,0);

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

}

**Numerical Method**

**Simpson Integration ( ?? )**

double F(double x){

//被积函数

return sqrt(1+4\*a\*a\*x\*x);

}

double simpson(double l,double r){

double m=(r+l)/2.0;

return (F(l)+F(r)+4\*F(m))\*(r-l)/6.0;

}

double asr(double l,double r,double epsilon,double A){

double m=(r+l)/2.0;

double L=simpson(l,m),R=simpson(m,r);

if(fabs(L+R-A)<15\*epsilon)return L+R+(L+R-A)/15.0;

return asr(l,m,epsilon/2.0,L)+asr(m,r,epsilon/2.0,R);

}

double asr(double l,double r,double epsilon){

return asr(l,r,epsilon,simpson(l,r));

}

double calc(double w,double h){

a=4.0\*h/w/w;

//上边界,下边界,精度控制

return asr(0,w/2,eps)\*2;

}

**Newton’s Method ( Fast, but quite useless.. )**

double calc(double theta){

double res=1.0;

//每个零点的一个范围内的所有数都能够迭代到该零点..

while(1){

double y=func(theta,res);

if(fabs(y)<eps)break;

else res=res-y/deriv(theta,res);

}

return res;

}

**Number Theory**

**Basics .. 线性筛(N) + 快速幂(lgN) + 扩展欧几里得(lgN) + 逆元(lgN) + 分解质因数(NsqrtN) ..**

void prime\_table(){

memset(is\_prime,1,sizeof(is\_prime));

for(int i=2;i<maxn;i++){

if(is\_prime[i])

prime[++cnt]=i;

for(int j=1;j<=cnt && i\*prime[j]<maxn;j++){

is\_prime[i\*prime[j]]=false;

if(i%prime[j]==0)break;

}

}

}//线性筛

void prime(ll x){

for(int i=2;i\*i<=x;i++)if(x%i==0){

p[++cnt][1]=i;

while(x%i==0)p[cnt][2]++,x/=i;

}

if(x>1)p[++cnt][1]=x,p[cnt][2]=1;

}//分解质因数

ll pow\_mod(ll x,ll y){

ll res=1;

while(y){

if(y&1)res\*=x,res%=mod;

x\*=x,x%=mod,y>>=1;

}

return res%mod;

}//快速幂

void ext\_gcd(ll a,ll b,ll &d,ll &x,ll &y){

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

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

}//扩展欧几里得

ll inv(ll a,ll c){

ll d,x,y;

ext\_gcd(a,c,d,x,y);

return (x+c)%c;

}//逆元

void phi\_table(){

phi[1]=1;

for(int i=2;i<maxn;i++){

if(!phi[i])phi[i]=i-1,prime[++cnt]=i;

for(int j=1;j<=cnt&&i\*prime[j]<maxn;j++){

if(i%prime[j]==0){phi[i\*prime[j]]=phi[i]\*prime[j];break;}

else phi[i\*prime[j]]=phi[i]\*(prime[j]-1);

}

}

}//线性筛

**Discrete Logarithm**

**baby step, giant step ( hash\_map\*sqrtN )**

int gcd(int x,int y){return !y?x:gcd(y,x%y);}

struct edge{int next,v,id;}e[mod];

int first[mod],cnt;

void add\_hash(int u,int num,int id){

e[++cnt].next=first[u],first[u]=cnt,e[cnt].v=num,e[cnt].id=id;

}

int \_hash(ll x){

int res=inf;

for(int i=first[x%mod];i!=-1;i=e[i].next)

if(e[i].v==x)res=min(res,e[i].id);

return res==inf?0:res;

}

void ext\_gcd(ll a,ll b,ll &d,ll &x,ll &y){

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

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

}

ll inv(ll a,ll c){

ll d,x,y;

ext\_gcd(a,c,d,x,y);

return (x+c)%c;

}

ll pow\_mod(ll x,ll y,ll mm){

ll res=1;

while(y){

if(y&1)res\*=x,res%=mm;

x\*=x,x%=mm,y>>=1;

}

return res%mm;

}

ll baby\_step(ll a,ll b,ll c){//a^x = b (mod c)

ll tt=1,tmp,co=0,d=1;

memset(first,-1,sizeof(first)),cnt=-1;

for(int i=0;i<=100;tt=tt\*a%c,i++)if(tt==b)return i;

while((tmp=gcd(a,c))!=1){

if(b%tmp!=0)return -1;

co++,b/=tmp,c/=tmp,d=d\*a/tmp%c;

}

int m=(int)(sqrt(c+0.5));

tt=1;

for(int i=0;i<=m;i++)

add\_hash(tt%mod,tt,i),tt\*=a,tt%=c;

tt=pow\_mod(a,m,c);

for(int i=0;i<=m;i++){

if(tmp=\_hash(b\*inv(d,c)%c))return i\*m+tmp+co;

d=d\*tt%c;

}

return -1;

}

**Chinese remainder theorem (NlgN)**

ll China(int N,ll a[],ll b[]){

ll d,x,y;

a[0]=1,b[0]=1;

for(int i=1;i<=N;i++){

ext\_gcd(a[i-1],a[i],d,x,y);

if((b[i]-b[i-1])%d!=0)return -1;

x\*=(b[i]-b[i-1])/d;

x%=(a[i]/d);if(x<0)x+=a[i]/d;

int lcm=a[i]\*a[i-1]/gcd(a[i],a[i-1]);

a[i]=lcm;

b[i]=(a[i-1]\*x+b[i-1])%lcm;

}

return b[N];

}//N mod a[i] = b[i]

**Lucas Theorem**

ll calc(ll x,ll y,ll c){

if(x<y)return 0;

ll res1=1,res2=1;

for(int i=1;i<=y;i++)

res1\*=(x-i+1),res1%=c,res2\*=i,res2%=c;

res2=inv(res2,c),res1\*=res2,res1%=c;

return res1;

}//C(x,y)%c,直接算

ll lucas(ll x,ll y,ll c){

ll res=1;

while(x&&y&&res)res\*=calc(x%c,y%c,c),res%=c,x/=c,y/=c;

return res;

}//C(x,y)%c,Lucas定理

**Linear module equation**

ll equ(ll a,ll b,ll c){

// ax + cy = b

ll d,x,y;ext\_gcd(a,c,d,x,y);

if(b%d!=0)return -1;

x\*=(b/d),x%=(c/d);if(x<0)x+=(c/d);

return x;

}//求解模方程ax = b mod c, 最小满足要求的正整数x

**Quadratic residue**

// x\*x == a (mod n) n should be a prime and gcd(a,n) == 1

// by discover

ll sqrt\_mod(ll a,ll n){

ll ret;

if(!a || n==2 || n==1)return a%n;

if(pow\_mod(a,(n-1)/2,n)==1){

if(n%4==3)ret=pow\_mod(a,(n+1)/4,n);

else{

ll b=1,k=0,i=(n-1)/2;

while(pow\_mod(b,(n-1)/2,n)==1)b++;

do{

i/=2,k/=2;

if((pow\_mod(a,i,n)\*(ll)pow\_mod(b,k,n)+1==0)%n)

k+=(n-1)/2;

}while(i%2==0);

ret=(pow\_mod(a,(i+1)/2,n)\*(ll)pow\_mod(b,k/2,n))%n;

}

return min(ret,n-ret);// make that res <= n/2

}

else return -1;

}

**Few other math techniques**

**Polya/Burnside ( Cube, edge+point+face )**

int main(){

cin>>t;

for(int cas=1;cas<=t;cas++){

scanf("%d",&n);//n -> color number

ans=pow\_mod(n,74);

ans+=2\*3\*pow\_mod(n,2+3+15);//face,90,270

ans+=3\*pow\_mod(n,4+6+28);//face,180

ans+=6\*pow\_mod(n,4+7+27);//edge,180

ans+=2\*4\*pow\_mod(n,4+4+18);//pnt,120,240

ans%=mod;

ll p=inv(24,mod);

printf("Case %d: %I64d\n",cas,(ans\*p)%mod);

}

return 0;

}

**FFT ( nlgn .. from cxlove )**

struct Complex{

double a,b;

Complex(double \_a=0.0,double \_b=0.0):a(\_a),b(\_b){}

Complex operator + (const Complex &c) const {return Complex(a + c.a , b + c.b);}

Complex operator - (const Complex &c) const {return Complex(a - c.a , b - c.b);}

Complex operator \* (const Complex &c) const {return Complex(a \* c.a - b \* c.b , a \* c.b + b \* c.a);}

};

void change(Complex y[],int len) {

for (int i=1,j=len/2;i<len-1;i++) {

if (i<j)swap(y[i],y[j]);

int k=len/2;

while(j>=k)

j-=k,k/=2;

if(j<k)j+=k;

}

}

void fft(int len,int on,Complex y[]){

change(y,len);

for (int h=2;h<=len;h<<=1) {

Complex wn(cos(-on\*2\*pi/h),sin(-on\*2\*pi/h));

for (int j=0;j<len;j+=h){

Complex w(1,0);

for (int k=j;k<j+h/2;k++) {

Complex u=y[k];

Complex t=w\*y[k+h/2];

y[k]=u+t,y[k+h/2]=u-t,w=w\*wn;

}

}

}

if(on==-1)

for(int i=0;i<len;i++)

y[i].a=(y[i].a/len);

}

Complex a[100005],b[100005];

int main{

int len=1;

while(len<2\*(mx+1))len<<=1;//注意是 mx+1 ..

for(int i=0;i<=len;i++)

a[i]=Complex(0,0),b[i]=Complex(0,0);

for(int i=mn;i<=mx;i++)

a[i]=Complex(before[i],0),

b[i]=Complex(after[i],0);

fft(len,1,a),fft(len,1,b);

for(int i=0;i<len;i++)a[i]=a[i]\*b[i];

fft(len,-1,a);

for(int i=st;i<=ed;i++)

res+=(ll)(a[2\*x[i]].a+0.5);

}

**SqrtN Partition addition**

for(int i=1;i<=min(a,b);i++){

int m=min(a/(a/i),b/(b/i))-i;

ans+=(sump[i+m]-sump[i-1])\*(a/i)\*(b/i);

i+=m;

}

**Miller\_rabin + Pollard\_rho**

ll witness(ll a,ll b,ll c){

if(b==0)return 1;

ll x,y,t=0;

while((b&1)==0)

b>>=1,t++;

y=x=pow\_mod(a,b,c);

while(t--){

y=mul\_mod(x,x,c);

if(y==1 && x!=1 && x!=c-1)

return false;

x=y;

}

return y==1;

}

bool miller\_rabin(ll n) {//..质数为true, 非质数为false..

if(n==2)return true;

if(n<2 || (n&1)==0)return false;

for(int i=0;i<3;i++)

if(witness(rand()%(n-2)+2,n-1,n)!=1)

return false;

return true;

}

ll pollard\_rho(ll n,ll c){//..随机返回一个 n 的约数..

if(n%2==0)return 2;

ll i=1,k=2,x=rand()%n,y=x,d;

while(1){

i++;

x=(mul\_mod(x,x,n)+c)%n;

d=gcd(y-x,n);

if(d==n)return n;

if(d!=n && d>1)return d;

if(i==k) y=x,k<<=1;

}

}

void calc(ll n,ll c=240){//寻找最小的约数..

if(n==1)return;

if(miller\_rabin(n)){

ans=min(ans,n);

return;

}

ll k=n;

while(k==n)k=pollard\_rho(n,c--);

calc(k,c),calc(n/k,c);

}

**Others**

**Manacher Algorithm**

#include <iostream>

#include <stdio.h>

#include <string.h>

#define maxn 200010

using namespace std;

int T,N,length[maxn],ans,str[maxn],c[maxn],p[maxn],cnt;

void gogo(int n){

//首先在每两个字符中间(包括首尾)插入一个没有出现过的字符,p[i]-1即是答案.

int mx=0,id;

for(int i=1;i<=n;i++){

if(mx>i)p[i]=min(p[2\*id-i],mx-i);

else p[i]=1;

for(;str[i+p[i]]==str[i-p[i]];p[i]++);

if(p[i]+i>mx)

mx =p[i]+i,id=i;

}

}

int main()

{

cin>>T;

while(T--){

ans=cnt=0;

scanf("%d",&N);

for(int i=1;i<=N;i++)

scanf("%d",&c[i]);

if(N==1){cout<<1<<endl;continue;}

length[N]=1;

for(int i=N-1;i>=1;i--){

if(c[i]>=c[i+1])length[i]=length[i+1]+1;

else length[i]=1;

}

str[0]=1;

for(int i=1;i<=N;i++)

str[++cnt]=c[i],str[++cnt]=1;

gogo(cnt);

for(int i=1;i<=cnt;i++){

if(!(i&1))

ans=max(ans,min(p[i]-1,length[i/2+1]\*2));

else

ans=max(ans,min(p[i]-1,length[i/2+1]\*2-1));

}

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

}

return 0;

}

**KMP**

int j=0;  
for(int i=1;i<=n;i++){  
 if(j>0&&b[j+1]!=a[i])j=p[j];  
 if(b[j+1]==a[i])j++;  
 if(j==m){cout<<i<<endl;j=p[j];}  
}  
//预处理B串  
int j=0;  
for(int i=2;i<=m;i++){  
 if(j>0&&b[j+1]!=b[i])j=p[j];  
 if(b[j+1]==b[i])j++;

p[i]=j;

}

**Smallest direct tree**

#include <iostream>

#include <stdio.h>

#include <vector>

#include <queue>

#include <string.h>

#include <math.h>

#define maxn 205

#define maxm 20005

#define inf 0x7fffffff

#define ll long long

#define pb push\_back

using namespace std;

struct pnt {double x,y;}a[maxn];

struct edge{int u,v,next;double c;}e[maxm];

int n,m,x,y,pre[maxn],cnt=-1,id[maxn],first[maxn],index,visit[maxn];

double mincost[maxn],ans;

bool flag[maxn];

double get\_dist(int p,int q){

return sqrt((a[p].x-a[q].x)\*(a[p].x-a[q].x)+(a[p].y-a[q].y)\*(a[p].y-a[q].y));

}

void addedge(int u,int v,double c){ e[++cnt].next=first[u],first[u]=cnt,e[cnt].v=v,e[cnt].u=u,e[cnt].c=c;

}

int dfs(int x){

flag[x]=true;

int num=1;

for(int i=first[x];i!=-1;i=e[i].next)if(!flag[e[i].v])

num+=dfs(e[i].v);

return num;

}

void directed\_mst(int root){

int size=n;

while(1){

for(int i=1;i<=size;i++)

mincost[i]=inf;

for(int i=0;i<=cnt;i++)

if(mincost[e[i].v]>e[i].c && e[i].u!=e[i].v){

mincost[e[i].v]=e[i].c;

pre[e[i].v]=e[i].u;

}

mincost[root]=0;

memset(id,0,sizeof(id));

memset(visit,0,sizeof(visit));

int index=0;

for(int i=1;i<=size;i++){

ans+=mincost[i];

int v=i;

while(v!=root && visit[v]!=i && !id[v])

visit[v]=i,v=pre[v];

if(v!=root && !id[v]){

id[v]=++index;

for(int u=pre[v];u!=v;u=pre[u])

id[u]=index;

}

}

if(!index)break;

for(int i=1;i<=size;i++)

if(!id[i])id[i]=++index;

for(int i=0;i<=cnt;i++){

int v=e[i].v;

e[i].u=id[e[i].u];

e[i].v=id[e[i].v];

if(e[i].u!=e[i].v)

e[i].c-=mincost[v];

}

size=index;

root=id[root];

}

printf("%.2lf\n",ans);

}

void init(){

cnt=-1,ans=0;

for(int i=1;i<=n;i++)

first[i]=-1,flag[i]=false;

}

int main(){

while(scanf("%d%d",&n,&m)!=EOF){

init();

for(int i=1;i<=n;i++)

scanf("%lf%lf",&a[i].x,&a[i].y);

for(int i=1;i<=m;i++){

scanf("%d%d",&x,&y);

if(x!=y)

addedge(x,y,get\_dist(x,y));

}

if(dfs(1)!=n)printf("poor snoopy\n");

else directed\_mst(1);

}

return 0;

}

**Kth short path**

#include <iostream>

#include <vector>

#include <queue>

#define MAXN 1005

#define MAXM 100005

#define inf 0x3fffffff

using namespace std;

struct AStar{int F,G,V;}TTemp;//G=F+V..V用dijkstra预处理

struct Edge{int V,Cost;}Temp;

int N,M,S,T,K,Dist[MAXN],Cnt[MAXN];

vector<Edge>E[MAXM],EE[MAXM];

bool operator < (const AStar &A,const AStar &B){return A.F>B.F;}

void Spfa(){

bool Flag[MAXN];

memset(Flag,true,sizeof(Flag));

for(int i=1;i<=N;i++)Dist[i]=inf;

queue<int>Q;

Q.push(T);Dist[T]=0;

while(!Q.empty()){

int F=Q.front();

Q.pop();

Flag[F]=true;

for(int i=0,SS=EE[F].size();i<SS;i++)

if(Dist[EE[F][i].V]>EE[F][i].Cost+Dist[F]){

Dist[EE[F][i].V]=EE[F][i].Cost+Dist[F];

if(Flag[EE[F][i].V]){

Q.push(EE[F][i].V);

Flag[EE[F][i].V]=false;

}

}

}

//for(int i=1;i<=N;i++)

// cout<<i<<' '<<Dist[i]<<endl;

}

int A\_Star(){

if(Dist[S]==inf)return -1;

priority\_queue <AStar> Q;

TTemp.F=Dist[S];TTemp.V=S;TTemp.G=0;

Q.push(TTemp);

while(!Q.empty()){

TTemp=Q.top();

Q.pop();

int F=TTemp.V;

//cout<<endl<<TTemp.V<<' '<<TTemp.G<<' '<<TTemp.F<<endl<<"-------------"<<endl;

Cnt[F]++;

if(Cnt[T]==K)return TTemp.G;

if(Cnt[F]>K)break;

for(int i=0,SS=E[F].size();i<SS;i++){

AStar AA;

AA.G=TTemp.G+E[F][i].Cost;

AA.F=AA.G+Dist[E[F][i].V];

AA.V=E[F][i].V;

//cout<<AA.V<<' '<<AA.G<<' '<<AA.F<<endl;

Q.push(AA);

}

}

return -1;

}

int main(){

cin>>N>>M;

int x,y,z;

for(int i=1;i<=M;i++){

scanf("%d%d%d",&x,&y,&z);

Temp.V=y,Temp.Cost=z;

E[x].push\_back(Temp);

Temp.V=x;EE[y].push\_back(Temp);

}

scanf("%d%d%d",&S,&T,&K);

if(S==T)K++;

Spfa();

cout<<A\_Star()<<endl;

system("pause");

return 0;

}

**MoDui’s Algorithm (NsqrtN)**

#include <iostream>

#include <stdio.h>

#include <math.h>

#include <string.h>

#include <vector>

#include <stdlib.h>

#include <map>

#include <queue>

#include <algorithm>

#pragma comment(linker, "/STACK:36777216")

#define lc(x) (x<<1)

#define rc(x) (lc(x)+1)

#define pi (acos(-1))

#define eps 1e-3

#define maxn 50005

#define maxm 10005

#define ll long long

#define inf 1e9

#define pb push\_back

#define mod 2007

using namespace std;

struct query{int l,r,id;ll ans1,ans2;}q[maxn];

ll num[maxn];

int n,m,a[maxn];

ll gcd(ll x,ll y){return !y?x:gcd(y,x%y);}

bool cmp1(const query &a,const query &b){

int k1=a.l/sqrt(n),k2=b.l/sqrt(n);

if(k1!=k2)return k1<k2;

return a.r<b.r;

}

bool cmp2(const query &a,const query &b){return a.id<b.id;}

int main(){

while(scanf("%d%d",&n,&m)==2){

memset(num,0,sizeof(num));

for(int i=1;i<=n;i++)

scanf("%d",&a[i]);

for(int i=1;i<=m;i++)

scanf("%d%d",&q[i].l,&q[i].r),q[i].id=i;

sort(q+1,q+1+m,cmp1);

int head=1,tail=1;num[a[1]]++;

ll res=0;

for(int i=1;i<=m;i++){

while(head<q[i].l)res-=--num[a[head]],head++;

while(head>q[i].l)head--,res+=num[a[head]]++;

while(tail<q[i].r)tail++,res+=num[a[tail]]++;

while(tail>q[i].r)res-=--num[a[tail]],tail--;

q[i].ans1=res,q[i].ans2=(ll)(q[i].r-q[i].l+1)\*(ll)(q[i].r-q[i].l)/2;

//cout<<q[i].l<<' '<<q[i].r<<' '<<res<<endl;

}

sort(q+1,q+1+m,cmp2);

for(int i=1;i<=m;i++){

ll tmp=gcd(q[i].ans1,q[i].ans2);

if(q[i].ans1==0)printf("0/1\n");

else printf("%lld/%lld\n",q[i].ans1/tmp,q[i].ans2/tmp);

}

}

}

**Merge Sort & Calculate Inverse Pairs**

#pragma comment(linker, "/STACK:36777216")

#include <map>

#include <set>

#include <cmath>

#include <queue>

#include <cstdio>

#include <string>

#include <vector>

#include <iomanip>

#include <cassert>

#include <cstdlib>

#include <cstring>

#include <iostream>

#include <algorithm>

#define lc(x) (x<<1)

#define rc(x) (lc(x)+1)

#define PI (acos(-1))

#define EPS 1e-8

#define MAXN 1111111

#define MAXM 888888

#define LL long long

#define ULL unsigned long long

#define INF 0x7fffffff

#define pb push\_back

#define mp make\_pair

#define MOD 1000000007

#define lowbit(x) (x&(-x))

using namespace std;

LL tot, sum[33], sum2[33];

int a[MAXN], b[MAXN], c[MAXN];

void gao(int dd, int l, int r){

if(dd == 0) return;

int mid = l + r >> 1;

gao(dd - 1, l, mid), gao(dd - 1, mid + 1, r);

vector <int> vec;

int lp = l, rp = mid + 1;

while(lp <= mid && rp <= r){

if(b[lp] <= b[rp]) vec.pb(b[lp]), lp ++, sum[dd] += (rp - mid - 1);

else vec.pb(b[rp]), rp ++;

}

while(lp <= mid) vec.pb(b[lp]), lp ++, sum[dd] += (r - mid);

while(rp <= r) vec.pb(b[rp]), rp ++;

for(int i = 0; i < vec.size(); i ++){

b[l + i] = vec[i];

//cout << l << ' ' << vec[i] << endl;

}

}

void gao2(int dd, int l, int r){

if(dd == 0) return;

int mid = l + r >> 1;

gao2(dd - 1, l, mid), gao2(dd - 1, mid + 1, r);

vector <int> vec;

int lp = l, rp = mid + 1;

while(lp <= mid && rp <= r){

if(c[lp] <= c[rp]) vec.pb(c[lp]), lp ++, sum2[dd] += (rp - mid - 1);

else vec.pb(c[rp]), rp ++;

}

while(lp <= mid) vec.pb(c[lp]), lp ++, sum2[dd] += (r - mid);

while(rp <= r) vec.pb(c[rp]), rp ++;

for(int i = 0; i < vec.size(); i ++){

c[l + i] = vec[i];

//cout << l << ' ' << vec[i] << endl;

}

}

int n, m, q, x;

int main(){

//freopen("in.txt", "r", stdin);

cin >> n;

m = 1;

for(int i = 1; i <= n; i ++) m \*= 2;

for(int i = 1; i <= m; i ++) scanf("%d", &a[i]), b[i] = a[i], c[m - i + 1] = a[i];

gao(n, 1, m), gao2(n, 1, m);

for(int i = 1; i <= n; i ++)

tot += sum[i];

//cout << sum[1] << ' ' << sum2[1] << endl;

cin >> q;

while(q --){

scanf("%d", &x);

for(int i = x; i >= 1; i --){

tot = tot - sum[i] + sum2[i];

swap(sum[i], sum2[i]);

}

printf("%I64d\n", tot);

}

}