1.树链剖分
2.主席树
3.sais
4.树套树
5.二维线段树
6.sa+倍增 下一位置不确定

1.树链剖分 点权

```
struct edge{int nxt,to;}e[maxn<<1];
int cnt,head[maxn];
int fa[maxn],deep[maxn],siz[maxn],son[maxn],rk[maxn],top[maxn],id[maxn];
void adde(int u,int v){
    e[++cnt].nxt=head[u];e[cnt].to=v;head[u]=cnt;
}
void dfs1(int u,int pre,int dep){
    fa[u]=pre;deep[u]=dep;siz[u]=1;
    for(int i=head[u];i;i=e[i].nxt){
        int v=e[i].to;
        if(v==pre)continue;
        dfs1(v,u,dep+1);
        siz[u]+=siz[v];
        if(siz[v]>siz[son[u]])son[u]=v;
    }
}
void dfs2(int u,int t){
    top[u]=t;id[u]=++cnt;rk[cnt]=u;
    if(!son[u])return;
    dfs2(son[u],t);
    for(int i=head[u];i;i=e[i].nxt){
        int v=e[i].to;
        if(v!=son[u]&&v!=fa[u])dfs2(v,v);
    }
}
/**************************/
int qquery(int x,int y){
    int ans=0;
    while(top[x]!=top[y]){
        if(deep[top[x]] < deep[top[y]])swap(x,y);
        int tmp=query(id[top[x]],id[x],1,n,1);
        if(tmp)ans=tmp;
        x=fa[top[x]];
    }
    if(deep[x]>deep[y])swap(x,y);
    int tmp=query(id[x],id[y],1,n,1);
    if(tmp)ans=tmp;
    return ans;
}
void uupdate(int x,int y){
    while(top[x]!=top[y]){
        if(deep[top[x]] < deep[top[y]])swap(x,y);
        update(id[top[x]],id[x],1,n,1);
```

```
        x=fa[top[x]];
    }
    if(deep[x]>deep[y])swap(x,y);
    update(id[x],id[y],1,n,1);
}
/*************************/
int u,v,w;
char op[10];
int main(){
    scanf("%d%d",&n,&q);
    for(int i=1;i<n;i++){
        scanf("%d%d",&u,&v);
        adde(u,v);adde(v,u);
    }
    rk[0]=-1;
    cnt=0;dfs1(1,0,1);dfs2(1,1);
    while(q--){
        scanf("%s",op);
        if(op[0]=='0'){
            scanf("%d",&u);
            uupdate(u,u);
        }else{
            scanf("%d",&u);
            printf("%d\n",rk[qquery(1,u)]);
        }
    }
    return 0;
}
int query_lca(int x,int y){
    while(top[x]!=top[y]){
        if(deep[top[x]]<deep[top[y]])swap(x,y);
        x=fa[top[x]];
    }return deep[x]<deep[y]?x:y;
}
int qquery(int x,int y){
    int ans=0;
    while(top[x]!=top[y]){
        if(deep[top[x]] < deep[top[y]])swap(x,y);
        ans=max(ans , query(id[top[x]],id[x],1,n,1));
        x=fa[top[x]];
    }
    if(x==y)return ans;
    if(deep[x]>deep[y])swap(x,y);
    ans = max(ans , query(id[son[x]],id[y],1,n,1));
```

```
    return ans;
}
void uupdate(int x,int y,int v){
    while(top[x]!=top[y]){
        if(deep[top[x]] < deep[top[y]])swap(x,y);
        update(id[top[x]],id[x],v,1,n,1);
        x=fa[top[x]];
    }
    if(x==y)return;
    if(deep[x]>deep[y])swap(x,y);
    update(id[son[x]],id[y],v,1,n,1);
}
```

2.可持久化线段树

```
const int maxn = 2e6+10;
int n,q,m,tot;
int a[maxn],t[maxn];
int T[maxn],lson[maxn*22],rson[maxn*22],c[maxn*22];
void init_hs(){
    for(int i=1;i<=n;i++){
        t[i]=a[i];
    }sort(t+1,t+1+n);
    m = unique(t+1,t+1+n)-t-1;
}
int build(int l,int r){
    int root=tot++;
    c[root]=0;
    if(l!=r){
        int mid=l+r>>1;
        lson[root]=build(l,mid);
        rson[root]=build(mid+1,r);
    }
    return root;
}
int hs(int x){
    return lower_bound(t+1,t+1+m,x)-t;
}
int update(int root,int pos,int val){
    int newroot=tot++,tmp=newroot;
    c[newroot]=val;
    int l=1,r=n;
    while(l<r){
        int mid=l+r>>1;
        if(pos<=mid){
```

```
            lson[newroot]=tot++;
            rson[newroot]=rson[root];
            newroot=lson[newroot];
            root=lson[root];
            r=mid;
        }
        else{
            rson[newroot]=tot++;
            lson[newroot]=lson[root];
            newroot=rson[newroot];
            root=rson[root];
            l=mid+1;
        }
        c[newroot]=val;
    }
    return tmp;
}
int query(int root,int id){
    int l=1,r=n;
    while(l<r){
        int mid=l+r>>1;
        if(id<=mid){
            r=mid;
            root=lson[root];
        }
        else{
            l=mid+1;
            root=rson[root];
        }
    }
    return c[root];
}
int v[maxn];
int main(){
    read(n,m);
    for(int i=1;i<=n;i++){
        read(a[i]);
    }
    T[n+1]=build(1,n);
    for(int i=n;i;i--){
        T[i]=update(T[i+1],i,a[i]);
    }
    v[0]=T[1];
    int op,vi,loci,valuei;
```

```
    for(int i=1;i<=m;i++){
        read(vi,op);
        if(op==1){
            read(loci,valuei);
            v[i]=update(v[vi],loci,valuei);
        }
        else{
            read(loci);
            v[i]=v[vi];
            print(query(v[i],loci),"\n");
        }
    }
    return 0;
}
```

区间修改历史版本线段树

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;
#define dlson l,mid,lson[nrt]
#define drson mid+1,r,rson[nrt]
const int maxn = 2e5+10;
int n,m;
int T[maxn],tot,lson[maxn<<5],rson[maxn<<5];
ll c[maxn<<5],cur[maxn<<5];
void push_up(int rt){
    c[rt]=c[lson[rt]]+c[rson[rt]];
}
void build(int l,int r,int &nrt){
    nrt=tot++;cur[nrt]=0;
    if(l==r){
        scanf("%lld",c+nrt);return;
    }int mid=l+r>>1;
    build(dlson);build(drson);push_up(nrt);
}
void update(int L,int R,ll val,int l,int r,int &nrt,int rt){
    nrt=tot++;
    lson[nrt]=lson[rt];
    rson[nrt]=rson[rt];
    c[nrt]=c[rt]+(R-L+1)*val;
    cur[nrt]=cur[rt];
    if(L==l&&R==r){
        cur[nrt]+=val;return;
    }
```

```
    int mid=l+r>>1;
    if(R<=mid)update(L,R,val,dlson,lson[rt]);
    else if(L>mid)update(L,R,val,drson,rson[rt]);
    else update(L,mid,val,dlson,lson[rt]),update(mid+1,R,val,drson,rson[rt]);
}
ll query(int L,int R,int l,int r,int nrt,ll add){
    if(L==l&&R==r)return c[nrt]+(R-L+1)*add;
    int mid=l+r>>1;
    if(R<=mid)return query(L,R,dlson,add+cur[nrt]);
    else if(L>mid)return query(L,R,drson,add+cur[nrt]);
    return query(L,mid,dlson,add+cur[nrt])+query(mid+1,R,drson,add+cur[nrt]);
}
int main(){
    while(scanf("%d%d",&n,&m)!=EOF){
    tot=0;
    build(1,n,T[0]);
    int time=0;
    char op[2];
    int l,r,d,t;
    for(int i=1;i<=m;i++){
        scanf("%s",op);
        if(op[0]=='C'){
            scanf("%d%d%d",&l,&r,&d);
            update(l,r,d,1,n,T[time+1],T[time]);time++;
        }
        else if(op[0]=='Q'){
            scanf("%d%d",&l,&r);
            printf("%lld\n",query(l,r,1,n,T[time],0));
        }
        else if(op[0]=='H'){
            scanf("%d%d%d",&l,&r,&t);
            printf("%lld\n",query(l,r,1,n,T[t],0));
        }
        else{
            scanf("%d",&t);
            tot=T[t+1];
            time=t;
        }
    }
    }
    return 0;
}
```

谈笑风生

```
#include <bits/stdc++.h>
using namespace std;
typedef long long ll;

const int maxn = 3e5+10;
int n,q;
int cnt,head[maxn];
int maxdep;
struct edge{int nxt,to;}e[maxn<<1];
int siz[maxn],dep[maxn],id[maxn],rk[maxn];
void adde(int u,int v){
    e[++cnt].nxt=head[u];
    e[cnt].to=v;
    head[u]=cnt;
}
void dfs(int u,int pre,int deep){
    maxdep=max(maxdep,deep);
    dep[u]=deep;siz[u]=1;id[u]=++cnt;rk[cnt]=u;
    for(int i=head[u];i;i=e[i].nxt){
        int v=e[i].to;
        if(v==pre)continue;
        dfs(v,u,deep+1);
        siz[u]+=siz[v];
    }
}
int T[maxn],lson[maxn<<5],rson[maxn<<5],TOT;
ll c[maxn<<5];
int build(int l,int r){
    int rt=TOT++;
    c[rt]=0;
    if(l!=r){
        int mid=l+r>>1;
        lson[rt]=build(l,mid);rson[rt]=build(mid+1,r);
    }return rt;
}
int update(int rt,int pos,int val){
    int nrt=TOT++,tmp=nrt;
    c[nrt]=c[rt]+val;
    int l=1,r=maxdep;
    while(l<r){
        int mid=l+r>>1;
        if(pos<=mid){
            lson[nrt]=TOT++,rson[nrt]=rson[rt];nrt=lson[nrt];rt=lson[rt];r=mid;
```

```
        }else{
            rson[nrt]=TOT++,lson[nrt]=lson[rt];nrt=rson[nrt];rt=rson[rt];l=mid+1;
        }c[nrt]=c[rt]+val;
    }return tmp;
}
ll query(int lrt,int rrt,int l,int r,int L,int R){
    if(L<=l&&R>=r){
        return c[rrt]-c[lrt];
    }int mid=l+r>>1;
    ll ret=0;
    if(L<=mid)ret+=query(lson[lrt],lson[rrt],l,mid,L,R);
    if(R>mid) ret+=query(rson[lrt],rson[rrt],mid+1,r,L,R);
    return ret;
}
int main(){
    scanf("%d%d",&n,&q);
    int u,v;
    for(int i=1;i<n;i++){
        scanf("%d%d",&u,&v);
        adde(u,v);adde(v,u);
    }
    cnt=0;dfs(1,0,1);
    T[0]=build(1,maxdep);
    for(int i=1;i<=n;i++){
        T[i]=update(T[i-1],dep[rk[i]],siz[rk[i]]-1);
    }
    while(q--){
        scanf("%d%d",&u,&v);
        if(dep[u]==maxdep){puts("0");continue;}
        ll ans=1ll*(siz[u]-1)*min(v,dep[u]-1);

ans+=query(T[id[u]],T[id[u]+siz[u]-1],1,maxdep,dep[u]+1,min(maxdep,dep[u]+v));
        cout<<ans<<endl;
    }
    return 0;
}
```

链上修改第 k 大

```
#include<bits/stdc++.h>
#define il inline
#define vd void
typedef long long ll;
```

```
il int gi(){
    int x=0,f=1;
    char ch=getchar();
    while(!isdigit(ch)){
        if(ch=='-')f=-1;
        ch=getchar();
    }
    while(isdigit(ch))x=x*10+ch-'0',ch=getchar();
    return x*f;
}
int n;
int W[80010],w[160010],m;
int K[80010],A[80010],B[80010],dep[80010];
int fir[80010],dis[160010],nxt[160010],id;
il vd link(int a,int b){nxt[++id]=fir[a],fir[a]=id,dis[id]=b;}
int dfn[80010],siz[80010];
int rt[80010],ls[10000010],rs[10000010],sum[10000010],cnt;
int st[17][80010];
il vd dfs(int x,int fa=-1){
    dfn[x]=++dfn[0];siz[x]=1;
    for(int i=fir[x];i;i=nxt[i]){
        if(dis[i]==fa)continue;
        st[0][dis[i]]=x;
        dep[dis[i]]=dep[x]+1;
        dfs(dis[i],x);
        siz[x]+=siz[dis[i]];
    }
}
#define mid ((l+r)>>1)
il vd update(int&x,int l,int r,const int&p,const int&d){
    if(!x)x=++cnt;sum[x]+=d;if(l==r)return;
    if(p<=mid)update(ls[x],l,mid,p,d);
    else update(rs[x],mid+1,r,p,d);
}
#undef mid
il vd Update(int l,int r,int w,int d){
    while(l<=n)update(rt[l],1,m,w,d),l+=l&-l;
    while(r<=n)update(rt[r],1,m,w,-d),r+=r&-r;
}
il int LCA(int x,int y){
    if(dep[x]<dep[y])std::swap(x,y);
    int C=dep[x]-dep[y];
    for(int i=16;~i;--i)if(C&(1<<i))x=st[i][x];
    for(int i=16;~i;--i)if(st[i][x]^st[i][y])x=st[i][x],y=st[i][y];
```

```
    if(x^y)x=st[0][x];return x;
}
int main(){
    n=gi();
    int q=gi(),x,y;
    for(int i=1;i<=n;++i)W[i]=w[++m]=gi();
    for(int i=1;i<n;++i)x=gi(),y=gi(),link(x,y),link(y,x);
    for(int i=1;i<=q;++i){
        K[i]=gi(),A[i]=gi(),B[i]=gi();
        if(!K[i])w[++m]=B[i];
    }
    std::sort(w+1,w+m+1),m=std::unique(w+1,w+m+1)-w-1;
    for(int i=1;i<=n;++i)W[i]=std::lower_bound(w+1,w+m+1,W[i])-w;
    for(int i=1;i<=q;++i)if(!K[i])B[i]=std::lower_bound(w+1,w+m+1,B[i])-w;
    dfs(1);
    for(int i=1;i<17;++i)
        for(int j=1;j<=n;++j)
            st[i][j]=st[i-1][st[i-1][j]];
    for(int i=1;i<=n;++i)Update(dfn[i],dfn[i]+siz[i],W[i],1);
    for(int i=1;i<=q;++i)
        if(K[i]){
            x=A[i],y=B[i];int lca=LCA(x,y);
            if(dep[x]+dep[y]-2*dep[lca]+1<K[i]){puts("invalid request!");continue;}
            static int A[2333],B[2333],a,b;
            a=b=0;
            int l,r;
            r=dfn[x];while(r)B[++b]=rt[r],r-=r&-r;
            r=dfn[y];while(r)B[++b]=rt[r],r-=r&-r;
            r=dfn[lca];while(r)A[++a]=rt[r],r-=r&-r;
            if(lca!=1){r=dfn[st[0][lca]];while(r)A[++a]=rt[r],r-=r&-r;}
            l=1,r=m;
            while(l^r){
                int tot=0;
                for(int i=1;i<=a;++i)tot-=sum[rs[A[i]]];
                for(int i=1;i<=b;++i)tot+=sum[rs[B[i]]];
                if(K[i]>tot){
                    K[i]-=tot;
                    for(int i=1;i<=a;++i)A[i]=ls[A[i]];
                    for(int i=1;i<=b;++i)B[i]=ls[B[i]];
                    r=(l+r)>>1;
                }else{
                    for(int i=1;i<=a;++i)A[i]=rs[A[i]];
                    for(int i=1;i<=b;++i)B[i]=rs[B[i]];
                    l=((l+r)>>1)+1;
```

```
            }
        }
        printf("%d\n",w[l]);
    }else
Update(dfn[A[i]],dfn[A[i]]+siz[A[i]],W[A[i]],-1),Update(dfn[A[i]],dfn[A[i]]+siz[A[i
]],W[A[i]]=B[i],1);
    return 0;
}
```

3.sais

```
*****************************************/
typedef long long ll;
template <typename T>
inline void gm(T*& bas, int siz, T*& op) {
    op = bas;
    bas += siz;// 这里把内存提前申请好然后用指针分配内存
}
//这里 sum 表示桶的位置，cur 是 sum 的一份 copy，避免每次都要做前缀和
// S 型后缀的桶是倒着开的也要倒着扫，L 型后缀的桶的是正着开的也要正着扫
#define pus(x) (sa[cur[a[x]]--] = x)  // 插入 S 型后缀
#define pul(x) (sa[cur[a[x]]++] = x)  //插入诱导排序
//诱导排序的宏，进行了两轮诱导过程:分别是从 lms 诱导到 L 和从 L 诱导到 S
#define inds(lms)                                       \
    for (int i = 1; i <= n; i++) sa[i] = -1;             \
    for (int i = 1; i <= n; i++) sum[i] = 0;             \
    for (int i = 1; i <= n; i++) sum[a[i]]++;            \
    for (int i = 1; i <= n; i++) sum[i] += sum[i - 1];    \
    for (int i = 1; i <= n; i++) cur[i] = sum[i];         \
    for (int i = m; i >= 1; i--) pus(lms[i]);            \
    for (int i = 1; i <= n; i++) cur[i] = sum[i - 1] + 1; \
    for (int i = 1; i <= n; i++)                          \
        if (sa[i] > 1 && !tp[sa[i] - 1])                  \
            pul(sa[i] - 1);                              \
    for (int i = 1; i <= n; i++) cur[i] = sum[i];         \
    for (int i = n; i >= 1; i--)                          \
        if (sa[i] > 1 && tp[sa[i] - 1])                   \
            pus(sa[i] - 1);
int sa[N];
int sum[N];
int cur[N];
int rk[N];
```

```
int A_bas[N << 4];
int* A_t;
inline void sais(int n, int* a) {
    int* tp;           //现开数组现分配内存
    gm(A_t, n + 1, tp);
    int* p;
    gm(A_t, n + 2, p);
    tp[n] = 1;
    //倒着扫一遍确定后缀类型
    for (int i = n - 1; i >= 1; i--) tp[i] = (a[i] == a[i + 1]) ? tp[i + 1] : (a[i] <
a[i + 1]);
    //确定第 i 个 lms 后缀的编号和编号为 i 的后缀是第几个 lms 后缀
    int m = 0;
    for (int i = 1; i <= n; i++) rk[i] = (tp[i] && !tp[i - 1]) ? (p[++m] = i, m) : -1;
    //进行一轮诱导排序，生成新的字符串
    inds(p);
    int tot = 0;
    int* a1;
    gm(A_t, m + 1, a1);
    p[m + 1] = n;
    //扫一遍对 lms 子串进行离散化
    for (int i = 1, x, y; i <= n; i++)
        if ((x = rk[sa[i]]) != -1) {
            if (tot == 0 || p[x + 1] - p[x] != p[y + 1] - p[y]) //如果长度不一致显然不
等
                tot++;
            //否则暴力 for 一遍比较两个 lms 串是否相等
            else
                for (int p1 = p[x], p2 = p[y]; p2 <= p[y + 1]; p1++, p2++)
                    if ((a[p1] << 1 | tp[p1]) != (a[p2] << 1 | tp[p2])) {
                        tot++;
                        break;
                    }
            a1[y = x] = tot;
        }
    if (tot == m)
        for (int i = 1; i <= m; i++) sa[a1[i]] = i; //如果字符互不相同就直接计算后缀数组，
否则递归
    else
        sais(m, a1);
    for (int i = 1; i <= m; i++) a1[i] = p[sa[i]]; //还原 lms 子串的顺序，进行诱导排序
    inds(a1);
}
char mde[N];
```

```
int n;
int a[N];
int tr[300];
char buf[20];
int cnt;
int main() {
    A_t = A_bas;
    scanf("%s", mde + 1);
    while (mde[n + 1] != '\0') n++;
    for (int i = 1; i <= n; i++) tr[mde[i]] = 1;
    for (int i = 1; i < 300; i++) tr[i] += tr[i - 1];
    for (int i = 1; i <= n; i++) a[i] = tr[mde[i]] + 1;
    a[++n] = 1;
    sais(n, a);
    for (int i = 2; i <= n; i++) {
        int tmp = sa[i];
        while (tmp) buf[++cnt] = tmp % 10 + 48, tmp /= 10;
        while (cnt) putchar(buf[cnt--]);
        putchar(' ');
    }
    return 0;  //拜拜程序~
}
```

4.seg + splay

```
#include<cstdio>
#include<cmath>
#include<string>
#include<iostream>
#include<algorithm>
#define ll long long
#define RI register int
#define A printf("A")
#define C printf(" ")
#define inf 2147483647
#define PI 3.1415926535898
using namespace std;
const int N=4e6+2;
//template <typename _Tp> inline _Tp max(const _Tp&x,const _Tp&y){return x>y?x:y;}
//template <typename _Tp> inline _Tp min(const _Tp&x,const _Tp&y){return x<y?x:y;}
template <typename _Tp> inline void IN(_Tp&x){
    char ch;bool flag=0;x=0;
    while(ch=getchar(),!isdigit(ch))if(ch=='-')flag=1;
    while(isdigit(ch))x=x*10+ch-'0',ch=getchar();
    if(flag)x=-x;
```

```
}
int n,m,a[N],ans,MX;
/*----------------------------------Splay-------------------------------------*/
int f[N],c[N],s[N],v[N],ch[N][2],rt[N],tot;
inline int chk(int x){return ch[f[x]][1]==x;};
inline void Splay_del_node(int x){f[x]=s[x]=c[x]=v[x]=ch[x][0]=ch[x][1]=0;};
inline                              void                              Splay_pushup(int
x){s[x]=(ch[x][0]?s[ch[x][0]]:0)+(ch[x][1]?s[ch[x][1]]:0)+c[x];};
inline void Splay_rotate(int x){
    int y=f[x],z=f[y],k=chk(x),v=ch[x][k^1];
    ch[y][k]=v;if(v)f[v]=y;f[x]=z;if(z)ch[z][chk(y)]=x;
    f[y]=x,ch[x][k^1]=y;Splay_pushup(y),Splay_pushup(x);
};
inline void Splay(int i,int x,int top=0){
     while(f[x]!=top){
        int y=f[x],z=f[y];
        if(z!=top)Splay_rotate((ch[z][0]==y)==(ch[y][0]==x)?y:x);
        Splay_rotate(x);
    }if(!top)rt[i]=x;
};
inline void Splay_Insert(int i,int x){
    int pos=rt[i];
    if(!rt[i]){
        rt[i]=pos=++tot;v[pos]=x;s[pos]=c[pos]=1;
        f[pos]=ch[pos][0]=ch[pos][1]=0;return;
    }int last=0;
    while(1){
        if(v[pos]==x){++c[pos];Splay_pushup(last);break;}
        last=pos;pos=ch[pos][x>v[pos]];
        if(!pos){
            pos=++tot;v[pos]=x;s[pos]=c[pos]=1;
            ch[last][x>v[last]]=pos;
            f[pos]=last;ch[pos][0]=ch[pos][1]=0;
            Splay_pushup(last);break;
        }
    }Splay(i,pos);return;
};
inline int Splay_rank(int i,int k){
    int x=rt[i],cal=0;
    while(x){
        if(v[x]==k)return cal+((ch[x][0])?s[ch[x][0]]:0);
        else if(v[x]<k){
            cal+=((ch[x][0])?s[ch[x][0]]:0)+c[x];x=ch[x][1];
        }else x=ch[x][0];
```

```
    }return cal;
};
inline int Splay_find(int i,int x){
    int pos=rt[i];while(x){
        if(v[pos]==x){Splay(i,pos);return pos;};
        pos=ch[pos][x>v[pos]];
    }return 0;
};
inline int Splay_pre(int i){int x=ch[rt[i]][0];while(ch[x][1])x=ch[x][1];return x;}
inline int Splay_suc(int i){int x=ch[rt[i]][1];while(ch[x][0])x=ch[x][0];return x;}
inline int Splay_Get_pre(int i,int x){
    int pos=rt[i];while(pos){
        if(v[pos]<x){if(ans<v[pos])ans=v[pos];pos=ch[pos][1];}
        else pos=ch[pos][0];
    }return ans;
};
inline int Splay_Get_suc(int i,int x){
    int pos=rt[i];while(pos){
        if(v[pos]>x){if(ans>v[pos])ans=v[pos];pos=ch[pos][0];}
        else pos=ch[pos][1];
    }return ans;
};
inline void Splay_Delete(int i,int key){
    int x=Splay_find(i,key);
    if(c[x]>1){--c[x];Splay_pushup(x);return;}
    if(!ch[x][0]&&!ch[x][1]){Splay_del_node(rt[i]);rt[i]=0;return;}
    if(!ch[x][0]){int y=ch[x][1];rt[i]=y;f[y]=0;return;}
    if(!ch[x][1]){int y=ch[x][0];rt[i]=y;f[y]=0;return;}
    int p=Splay_pre(i);int lastrt=rt[i];
    Splay(i,p,0);ch[rt[i]][1]=ch[lastrt][1];f[ch[lastrt][1]]=rt[i];
    Splay_del_node(lastrt);Splay_pushup(rt[i]);
};
/*------------------------------Seg_Tree--------------------------------------*/
#define lc ((x)<<1)
#define rc ((x)<<1|1)
#define mid ((l+r)>>1)
inline void Seg_Insert(int x,int l,int r,int pos,int val){
    Splay_Insert(x,val);if(l==r)return;
    if(pos<=mid)Seg_Insert(lc,l,mid,pos,val);
    else Seg_Insert(rc,mid+1,r,pos,val);
};
inline void Seg_rank(int x,int l,int r,int L,int R,int Kth){
    if(l==L&&r==R){ans+=Splay_rank(x,Kth);return;}
    if(R<=mid)Seg_rank(lc,l,mid,L,R,Kth);
```

```
    else if(L>mid)Seg_rank(rc,mid+1,r,L,R,Kth);
    else Seg_rank(lc,l,mid,L,mid,Kth),Seg_rank(rc,mid+1,r,mid+1,R,Kth);
};
inline void Seg_change(int x,int l,int r,int pos,int val){
//    printf("QvQ:: %d %d %d %d %d\n",x,l,r,pos,val);
    Splay_Delete(x,a[pos]);Splay_Insert(x,val);
    if(l==r){a[pos]=val;return;};
    if(pos<=mid)Seg_change(lc,l,mid,pos,val);
    else Seg_change(rc,mid+1,r,pos,val);
};
inline void Seg_pre(int x,int l,int r,int L,int R,int val){
    if(l==L&&r==R){ans=max(ans,Splay_Get_pre(x,val));return;}
    if(R<=mid)Seg_pre(lc,l,mid,L,R,val);
    else if(L>mid)Seg_pre(rc,mid+1,r,L,R,val);
    else Seg_pre(lc,l,mid,L,mid,val),Seg_pre(rc,mid+1,r,mid+1,R,val);
};
inline void Seg_suc(int x,int l,int r,int L,int R,int val){
    if(l==L&&r==R){ans=min(ans,Splay_Get_suc(x,val));return;}
    if(R<=mid)Seg_suc(lc,l,mid,L,R,val);
    else if(L>mid)Seg_suc(rc,mid+1,r,L,R,val);
    else Seg_suc(lc,l,mid,L,mid,val),Seg_suc(rc,mid+1,r,mid+1,R,val);
};
/*--------------------------------ask---------------------------------------*/
inline int Get_Kth(int x,int y,int k){
    int L=0,R=MX+1,M;
    while(L<R){
        M=(L+R)>>1;
        ans=0;Seg_rank(1,1,n,x,y,M);
        if(ans<k)L=M+1;else R=M;
    }return L-1;
};
/*-------------------------------main-------------------------------------*/
int main(int argc,char const* argv[]){
    IN(n),IN(m);
    for(RI i=1;i<=n;++i){IN(a[i]);Seg_Insert(1,1,n,i,a[i]);MX=max(MX,a[i]);}
    while(m--){
        int op,x,y,v;IN(op),IN(x),IN(y);
        switch(op){
            case 1:{IN(v);ans=0;Seg_rank(1,1,n,x,y,v);printf("%d\n",ans+1);}break;
            case 2:{IN(v);printf("%d\n",Get_Kth(x,y,v));}break;
            case 3:{Seg_change(1,1,n,x,y);}break;
            case 4:{IN(v);ans=-inf;Seg_pre(1,1,n,x,y,v);printf("%d\n",ans);}break;
            case 5:{IN(v);ans=inf;Seg_suc(1,1,n,x,y,v);printf("%d\n",ans);}break;
        }
```

```
    }return 0;
}
```

5.二维线段树

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;

#define lson l,mid,rt<<1
#define rson mid+1,r,rt<<1|1
const int maxn = 1e4+10;

int sum[maxn][maxn<<2];
char op[10];
int h;
double a,l;
int h1,h2;
double a1,a2;

void push_up(int rtx,int rty){
    sum[rtx][rty]=max(sum[rtx][rty<<1],sum[rtx][rty<<1|1]);
}
void buildy(int rtx,int l,int r,int rty){
    sum[rtx][rty]=-1;
    if(l==r)return;
    int mid=l+r>>1;
    buildy(rtx,l,mid,rty<<1);buildy(rtx,mid+1,r,rty<<1|1);
}
void build(int l,int r,int rtx){
    buildy(rtx,0,1000,1);
    int mid=l+r>>1;
    if(l!=r){
        build(l,mid,rtx<<1);build(mid+1,r,rtx<<1|1);
    }
}
void updatey(int rtx,int p,int v,int l,int r,int rty){
    if(l==r){
        sum[rtx][rty]=max(sum[rtx][rty],v);
        return;
    }
    int mid=l+r>>1;
    if(p<=mid)updatey(rtx,p,v,l,mid,rty<<1);
    else     updatey(rtx,p,v,mid+1,r,rty<<1|1);
    push_up(rtx,rty);
```

```
}
void update(int x,int y,int v,int l,int r,int rtx){
    updatey(rtx,y,v,0,1000,1);
    if(l!=r){
        int mid=l+r>>1;
        if(x<=mid)update(x,y,v,l,mid,rtx<<1);
        else      update(x,y,v,mid+1,r,rtx<<1|1);
    }
}
int queryy(int rtx,int L,int R,int l,int r,int rty){
    if(L<=l&&R>=r){
            return sum[rtx][rty];
    }
    int mid=l+r>>1;
    int res=-1;
    if(L<=mid)res=max(res,queryy(rtx,L,R,l,mid,rty<<1));
    if(R>mid)res=max(res,queryy(rtx,L,R,mid+1,r,rty<<1|1));
    return res;
}
int query(int Lx,int Rx,int Ly,int Ry,int l,int r,int rtx){
    if(Lx<=l&&Rx>=r)return queryy(rtx,Ly,Ry,0,1000,1);
    int mid=l+r>>1;
    int res=-1;
    if(Lx<=mid)res=max(res,query(Lx,Rx,Ly,Ry,l,mid,rtx<<1));
    if(Rx>mid)res=max(res,query(Lx,Rx,Ly,Ry,mid+1,r,rtx<<1|1));
    return res;

}
int main(){
    int m;
    while(scanf("%d",&m)!=EOF,m){
        while(m--){
            build(100,200,1);
            scanf("%s",op);
            if(op[0]=='I'){
                scanf("%d%lf%lf",&h,&a,&l);
                update(h,a*10,l*10,100,200,1);
            }
            else{
                scanf("%d%d%lf%lf",&h1,&h2,&a1,&a2);
                if(h1>h2)swap(h1,h2);
                if(a1>a2)swap(a1,a2);
                int ans=query(h1,h2,a1*10,a2*10,100,200,1);
                if(ans==-1)puts("-1");
```

```
            else printf("%.1f\n",0.1*ans);
        }
    }
}
return 0;
}
```

6.sa 下一位置不确定

```
#include<bits/stdc++.h>
using namespace std;
const int maxn=2e5;
char s[maxn+50];
int sa[maxn+50],rk[maxn+50];
int t[maxn+50],t2[maxn+50],c[maxn+50];
int nx[maxn+5][19];
int len,k;
queue<int> q[maxn+5];
void getsa(int m)//m 表示最大字符的编码
{
    memset(t,-1,sizeof(t));
    memset(t2,-1,sizeof(t2));
    int *x=t,*y=t2;
    for(int i=0;i<m;++i) c[i]=0;
    for(int i=0;i<len;++i) c[x[i]=s[i]]++;
    for(int i=1;i<m;++i) c[i]+=c[i-1];
    for(int i=len-1;i>=0;--i) sa[--c[x[i]]]=i;
    for(int j=0,k=1;k<=len;k<<=1,++j)
    {
        /*int p=0;
        for(int i=len-k;i<len;++i) y[p++]=i;
        for(int i=0;i<len;++i) if(sa[i]>=k) y[p++]=sa[i]-k;*/

        int p=0;
        for(int i=0;i<len;++i) q[nx[i][j]].push(i);
        for(int i=0;i<len;++i)
            while(!q[sa[i]].empty())
            {
                y[p++]=q[sa[i]].front();
                q[sa[i]].pop();
            }

        for(int i=0;i<m;++i) c[i]=0;
        for(int i=0;i<len;++i) c[x[y[i]]]++;
        for(int i=0;i<m;++i) c[i]+=c[i-1];
```

```
        for(int i=len-1;i>=0;--i) sa[--c[x[y[i]]]]=y[i];
        swap(x,y);
        p=1,x[sa[0]]=0;
        for(int i=1;i<len;++i)
            if(y[sa[i-1]]==y[sa[i]]&&y[nx[sa[i-1]][j]]==y[nx[sa[i]][j]])
x[sa[i]]=p-1;else x[sa[i]]=p++;
        if(p>=len) break;
        m=p;
    }
}
int main()
{
    int T;
    scanf("%d",&T);
    for(int cas=1;cas<=T;++cas)
    {
        printf("Case #%d: ",cas);
        scanf("%d",&len);
        scanf("%s",s);
        for(int i=0;i<len;++i) nx[i][0]=(1LL*i*i+1)%len;
        for(int j=1;j<=18;++j)
            for(int i=0;i<len;++i) nx[i][j]=nx[nx[i][j-1]][j-1];
        getsa('9'+1);
        int pos=sa[len-1];
        for(int i=1;i<=len;++i,pos=nx[pos][0]) printf("%c",s[pos]);
        printf("\n");
    }

   // for(int i=0;i<n;++i) printf("%d ",sa[i]);printf("\n");
   // for(int i=0;i<n;++i) printf("%d ",rk[i]);printf("\n");
   // for(int i=0;i<n;++i) printf("%d ",height[i]);printf("\n");
    return 0;

}
```

2.1 可重叠最长重复子串

求出 height 数组的最大值即可，复杂度 O(n)

2.2 不可重叠最长重复子串

先对排序后的后缀分组，二分答案 k，每组后缀之间的 height 值不小于 k

那么，如果有一组满足：

对于每组，每个后缀的 sa 值的最大值和最小值之差不小于 k，说明存在。

应用：利用 height 值对后缀进行分组的方法。复杂度 O(nlogn)

2.3 至少出现 k 次的最长子串

二分答案，用 height 分组，对于任意组，后缀数量大于 k，则存在。

复杂度 O(nlogn)

2.2.2 不同子串个数

对所有后缀排序之后，每一次新加的后缀，都会产生新的 n-sa[k]+1 个新前缀。

但是其中有 height[k]个是重复的，所以后缀 k 会贡献：

n-sa[k]+1-height[k]个不同的子串。累加就是答案。复杂度 O(n)

2.2.3 回文串

不想看了，有马拉车算法。

2.2.4 连续重复子串

用 kmp 求最小循环节便可，不看了。

2.2.5 重复次数最多的连续重复子串

跳过

2.3.1 最长公共子串

拼接两个串，找到满足以下条件的 height[]即可：

height 满足不在同一个字符串中，且 height 值最大。

复杂度 O(n+m)

2.3.2 长度不小于 k 的公共子串的个数

复杂，不看。

/*---*/

1.求字符串所有不同子串个数

我们知道 SA 数组里面是排好序的所有后缀；

对于每一个后缀，找到所有 n 个前缀串，所有后缀的前缀串数量相加得结果；

有的后缀之间存在公共前缀，要减去的个数正好是 LCP 的长度，也就是 height 数组。

```
int tot=0;
for(int i=0;i<n;i++){
    tot+=n-i-height[sa[i]];
}
```

2.求字符串中字典序第 k 大的子串

```
int tot=0;
int end;
for(int i=0;i<n;i++){
    int acc=n-sa[i]-height[i];
    if(tot+acc>=k){
        end = k-tot+height[i]-1;
        break;
    }tot+=acc;
}
```