sq

```
#include<iostream>
#include<algorithm>
#include<cstring>
#define int long long;
#include<unordered_map>
using namespace std;
int sq;
int n;
struct node
{
    int l,r,id;
    bool operator<(const node &o)const
    {
        if(l/sq!=o.l/sq)return l<o.l;
        if(l/sq&1)return r<o.r;
        return r>o.r;
    }
}qt[205000];
int a[205000];
unordered_map<int>mp;
struct point{
    int x,w;
}tmp;
void add(int p){
    mu[p]++;

}
void del(int p){
    mu[p]--;
}
int q;
int ans[205000];
void solve()
{
    int l=1,r=0;
    for(int i=1;i<=q;++i){
        while(q[i].l<l)add[--l];
        while(r<q[i].r)add[++r];
        while(q[i].l>l)del[l++];
        while(q[i].r<q[i].r)del[r--];
    }
}
signed main()
{
     std::ios::sync_with_stdio(false);
     cin>>n;
     for(int i=1;i<=n;++i)cin>>a[i];
     cin>>q;
     for(int i=1;i<=q;++i){
        cin>>qt[i].l>>qt[i].r;
```

```
        qt[i].id=i;
     }
     sort(qt+1,qt+n+1);
}
```

树

```
#include<iostream>
#include<cstring>
#include<string>
#include<algorithm>
#include<map>
#define int long long
using namespace std;
map<char,int>mp;
int trie[3000000+60][63],tag[3000000+60];
int tot;
char str[3000000+10];
void insert(char *s){
    int cur=0   ;
    int len=strlen(s+1);
    for(int i=1;i<=len;i++){
        if(!trie[cur][mp[s[i]]]){
            trie[cur][mp[s[i]]]=++tot;
        }
        cur=trie[cur][mp[s[i]]];
        tag[cur]++;
    }
}
int check(char* s){
    int cur=0;
    int len=strlen(str+1);
    for(int i=1;i<=len;i++){
        if(!trie[cur][mp[s[i]]])return 0;
        cur=trie[cur][mp[s[i]]];
    }
    return tag[cur];
}
signed main(){
    std::ios::sync_with_stdio(false);
    int t;
    cin>>t;
    int id=0;
    for(char i='a';i<='z';i++) mp[i]=++id;
    for(char i='A';i<='Z';i++) mp[i]=++id;
    for(char i='0';i<='9';i++) mp[i]=++id;
    while(t--){
        int n,m;
        cin>>n>>m;
        for(int i=0;i<=tot;++i){
            tag[i]=0;
            for(int j=0;j<=62;j++){
                trie[i][j]=0;
```

```
            }
        }
        tot=0;
        for(int i=1;i<=n;++i){
            cin>>str+1;
            insert(str);
        }
        for(int i=1;i<=m;++i){
            cin>>str+1;
            cout<<check(str)<<"\n";
        }
    }
}
```

dijstra

```
#include<iostream>
#include<algorithm>
#include<cstring>
#include<string>
#include<vector>
#include<queue>
#define int long long
using namespace std;
struct node
{
    int from,to,w;
}e[500050];
struct po
{
    int id,w;
    bool operator<(const po &o)const{
        return w>o.w;
    }
};
int nex[500050];
int cnt=0;
void add(int u,int v,int w){
    ++cnt;
    e[cnt].from=nex[u];
    e[cnt].to=v;
    nex[u]=cnt;
    e[cnt].w=w;
}
priority_queue<po>qu;
int dis[500005],book[500005];
int n,m,s;
void solve(int s){
    book[s]=1;
    for(int i=nex[s];i;i=e[i].from){
        dis[e[i].to]=min(dis[e[i].to],e[i].w);
    }
    for(int i=1;i<=n;++i){
        if(i==s||dis[i]==1234567890)continue;
        po tmp;
```

```
        tmp.w=dis[i];
        tmp.id=i;
        qu.push(tmp);
    }
    while(!qu.empty()){
        po tmp=qu.top();
        qu.pop();
        if(book[tmp.id])continue;
        book[tmp.id]=1;
        for(int i=nex[tmp.id];i;i=e[i].from){
            if(dis[e[i].to]>dis[tmp.id]+e[i].w){
                dis[e[i].to]=dis[tmp.id]+e[i].w;
                 po t;
                 t.id=e[i].to;
                 t.w=dis[e[i].to];
                 qu.push(t);
            }
        }
    }
    for(int i=1;i<=n;++i){
        if(dis[i]!=1234567890)
        cout<<dis[i]<<" ";
        else cout<<((1ll*1<<31)-1)<<" ";
    }
    cout<<"\n";

}
signed main(){

    std::ios::sync_with_stdio(false);cin>>n>>m>>s;
    for(int i=1;i<=n;++i)dis[i]=1234567890;
    dis[s]=0;
    for(int i=1;i<=m;++i){
        int u,v,w;
        cin>>u>>v>>w;
        add(u,v,w);
    }
    solve(s);

}
```

topo

```
#include<iostream>
#include<cstring>
#include<string>
#include<algorithm>
#include<queue>
#include<cstdio>
#include<string.h>
#define int long long
#define maxn 1000001
using namespace std;
struct kkk
{
```

```
    int son[26],flag,fail,ans;
    void clear(){for(int i=0;i<26;++i)son[i]=0;ans=fail=flag=0;}
}trie[2000000+1000];
int n,cnt,ans,in[2000000+1000],mp[2000000+1000];
queue<int>q;
char s[2000000+1000];
int vis[2000000+1000];
void insert(char *st,int num){
    int cur=1 ,len=strlen(st);
    for(int i=0;i<len;++i){
        int v=st[i]-'a';
        if(!trie[cur].son[v]){
            trie[cur].son[v]=++cnt;
        }
        cur=trie[cur].son[v];
    }
    if(!trie[cur].flag)
    trie[cur].flag=num;
    mp[num]=trie[cur].flag;
}
void get_fail(){
    for(int i=0;i<26;++i)trie[0].son[i]=1;
    q.push(1);trie[1].fail=0;
    while(!q.empty()){
        int u=q.front();q.pop();
        for(int i=0;i<26;++i){
            int v=trie[u].son[i];
            int fail=trie[u].fail;
            if(!v){
                trie[u].son[i]=trie[fail].son[i];
                continue;
            }
            trie[v].fail=trie[fail].son[i];
            in[trie[v].fail]++;
            q.push(v);
        }
    }
}
void topu(){
    for(int i=1;i<=cnt;++i){
        if(in[i]==0)q.push(i);
    }
    while(!q.empty()){
        int u=q.front();q.pop();
        vis[trie[u].flag]=trie[u].ans;
        int v=trie[u].fail;
        in[v]--;
        trie[v].ans+=trie[u].ans;
        if(!in[v])q.push(v);
    }
}
void query(char *s){
    int u=1,len=strlen(s);
    for(int i=0;i<len;++i){
        u=trie[u].son[s[i]-'a'],trie[u].ans++;
```

```
    }
}

signed main(){
    std::ios::sync_with_stdio(false);
    cin>>n;
    cnt =1;
    for(int i=1;i<=n;++i){
        cin>>s;
        insert(s,i);
    }
    cin>>s;
    get_fail();
    query(s);
    topu();
   for(int i=1;i<=n;++i)cout<<vis[mp[i]]<<"\n";
}
```