```cpp
//求N个字符串的至少在半数以上的字符串中出现过的最长公共子串并输出，如果有多条则排序后输出
#include <iostream>
#include <cstring>
#include <algorithm>
#include <cmath>
using namespace std;
const int maxn=200010;
int s[maxn];
int sa[maxn],t[maxn],t2[maxn],c[maxn],pos[maxn];
void build_sa(int n,int m){
    int i,*x=t,*y=t2;
    //初始化基数排序
    for (i=0;i<m;i++) c[i]=0;
    for (i=0;i<n;++i) ++c[ x[i]=s[i] ];
    for (i=1;i<m;++i) c[i]+=c[i-1];
    for (i=n-1;i>=0;--i) sa[ --c[x[i]] ]=i;
    //倍增基数排序
    for (int k=1;k<=n;k<<=1){
        int p=0;
        //先根据sa数组直接排序第二关键词
        for (i=n-k;i<n;++i) y[p++]=i;
        for (i=0;i<n;++i) if (sa[i]>=k) y[p++]=sa[i]-k;
        //再排序第一关键词
        for (i=0;i<m;i++) c[i]=0;
        for (i=0;i<n;++i) ++c[x[y[i]]];
        for (i=1;i<m;++i) c[i]+=c[i-1];
        for (i=n-1;i>=0;--i) sa[--c[x[y[i]]] ]=y[i];
        //根据sa数组重新构造x数组
        swap(x,y);
        p=1;x[sa[0]]=0;
        for (i=1;i<n;++i)
            x[sa[i]]= y[sa[i-1]]==y[sa[i]] && y[sa[i-1]+k]==y[sa[i]+k] ? p-1:p++;
        if (p>=n) break;
        m=p;
    }
}
int height[maxn],rank[maxn];
void getHeight(int n){
    int k=0;
    for (int i=0;i<n;++i) rank[sa[i]]=i;
    for (int i=0;i<n;++i) {
        if (k) k--;
        if (rank[i]){
            int j=sa[rank[i]-1];
            while(s[i+k]==s[j+k]) k++;
        }
        height[rank[i]]=k;
    }
}
bool judge(int n,int l,int k){
    int num=0;
    bool vis[k];
    memset(vis,false,sizeof(vis));
    for (int i=1;i<n;++i){
        if (height[i]<l) {
            memset(vis,false,sizeof(vis));
```

```
            num=0;
        }
        else{
            if (pos[sa[i]]==pos[sa[i-1]]) continue;
            if (!vis[pos[sa[i]]]) {
                vis[pos[sa[i]]]=true;
                num++;
            }
            if (!vis[pos[sa[i-1]]]) {
                vis[pos[sa[i-1]]]=true;
                num++;
            }
            if (num > (k/2) ) return true;
        }
    }
    return false;
}
bool print(int n,int l,int k){
    int num=0;
    bool vis[k];
    memset(vis,false,sizeof(vis));
    for (int i=0;i<n;++i){
        if (height[i]<l) {
            if (num>(k/2)) {
                for (int j=0;j<l;j++) cout<<char(s[j+sa[i-1]]+'a');
                cout<<endl;
            }
            memset(vis,false,sizeof(vis));
            num=0;
        }
        else{
            if (pos[sa[i]]==pos[sa[i-1]]) continue;
            if (!vis[pos[sa[i]]]) {
                vis[pos[sa[i]]]=true;
                num++;
            }
            if (!vis[pos[sa[i-1]]]) {
                vis[pos[sa[i-1]]]=true;
                num++;
            }
        }
    }
    return false;
}
int main(){
    //    freopen("datain.txt","r",stdin);
    ios::sync_with_stdio(false);
    cin.tie(0);
    int k;
    while(cin>>k && k){
        //读入预处理，pos数组用来存储一开始每个字母所在的字符串编号
        int n=0,tmp=0;
        for (int i=0;i<k;++i){
            string str;
            cin>>str;
            for (int j=0;str[j];++j) {
                pos[n]=tmp;
```

```
                s[n]=str[j]-'a';
                n++;
            }
            pos[n]=tmp;
            s[n]=30+(tmp++);
            n++;
        }
        //构造后缀数组
        build_sa(n,30+tmp+5);
        getHeight(n);
        //二分找到出现在半数以上的字符串中的最长公共子串的长度ans
        int l=0,r=n,ans=6;
        while(l<=r){
            int mid=(l+r)/2;
            if (judge(n,mid,k)) {
                ans=mid;
                l=mid+1;
            }
            else r=mid-1;
        }
        //输出
        if (ans) print(n,ans,k);
        else cout<<"?"<<endl;
        cout<<endl;
    }
}
```