由于所有的字符串长度都是相等的，因此最后的字符串必然可以切割成长度为m的部分，并且对称的部分只能是两个原本的字符串，或者是最为最终回文串的中间部分的那个字符串，它需要满足自己就是个回文串。

直接暴力匹配是否有对称的字符串就行了，注意中间还可以放一个自身就是回文串的字符串。

#include<bits/stdc++.h>

#define INF 0x3f3f3f3f //1061109567

#define llINF 9223372036854775807

#define pi 3.141592653589793

#define ll long long

**using** **namespace** std;

**const** ll maxn=1e2+7;

**const** **double** eps=1e-10;

**const** ll mod=1e9+7;

#define IOS ios::sync\_with\_stdio(0); cin.tie(0); cout.tie(0);

**bool** flag[maxn];

string s[maxn];

**int** main()

{

    IOS;

    ll n,m;

    cin>>n>>m;

**for**(ll i=1;i<=n;i++) cin>>s[i];

    string ans,mid,right;//ans暂存左侧部分，mid保存中间可能存在的对称部分，right保存右侧部分，也就是ans的镜像

**for**(ll i=1;i<=n;i++)

    {

**bool** F=0;   //F记录是否存在字符串与当前字符串镜像

**for**(ll j=i+1;j<=n;j++)

        {

**if**(!flag[j])    //如果该字符串仍然未被使用

            {

**bool** f=1;   //记录s[i]和s[j]是否为对称的字符串

**for**(ll k=0;k<m;k++)

**if**(s[i][k]!=s[j][m-k-1]) f=0;

**if**(f)   //如果满足

                {

                    flag[j]=1;  //使用字符串j

                    right=s[j]+right;   //将字符串j加到right部分去

                    F=1;        //记录找到s[i]的对称字符串，跳出循环

**break**;

                }

            }

        }

**if**(F)   ans+=s[i];  //如果找到了对称的字符串，那就把s[i]加到左侧的ans去

**else**    //如果没找到的话那就检测s[i]自身是不是一个回文串，如果是就放到mid部分去。

        {

            F=1;

**for**(ll j=0;j<m;j++)

**if**(s[i][j]!=s[i][m-j-1]) F=0;

**if**(F) mid=s[i];

        }

    }

    ans+=mid+right; //拼接ans+mid+right

    cout<<ans.size()<<endl;

    cout<<ans<<endl;

}