```
/* FILE: chargingChaos.cpp   last change: 30-Sep-2014   author: Romeo Rizzi
 * a solver for problem "chargingChaos"
 */

//#define NDEBUG   // NDEBUG definita nella versione che consegno
#include <cassert>
#ifndef NDEBUG
#  include <iostream>  // uso di cin e cout non consentito in versione finale
#endif
#include <fstream>
#include <algorithm>


using namespace std;

const int MAX_N = 100000;  int N;
const int MAX_L = 1000;  int L;

string source[MAX_N], target[MAX_N], flippedSource[MAX_N];
bool flip_position[MAX_L];


int main() {
  ifstream fin("input.txt"); assert( fin );
  fin >> N >> L; assert( N <= MAX_N ); assert( L <= MAX_L );

  for(int i = 0; i < N; i++)
    fin >> source[i];
  for(int i = 0; i < N; i++)
    fin >> target[i];
  fin.close();

  sort(target, target+N);

  int minMoves = L+1;
  for(int y = 0; y < N; y++) { /* we try to match the first element of the source on the y-th
element of the target */
    int yCost = 0;
    for(int j = 0; j < L; j++) {
       flip_position[j] = ( source[0][j] != target[y][j] );
       if( flip_position[j] ) yCost++;
    }
    if( yCost < minMoves ) {
       for(int i = 0; i < N; i++) {
          flippedSource[i] = source[i];
          for(int j = 0; j < L; j++)
            if( flip_position[j] )
               flippedSource[i][j] = (source[i][j] == '0')? '1' : '0';
       }

       sort(flippedSource, flippedSource+N);
       bool areEqual = true;
       for(int i = 0; i < N; i++)
         if( flippedSource[i] != target[i])  areEqual = false;
       if(areEqual)  minMoves = yCost;
    }
  }

  ofstream fout("output.txt"); assert( fout );
  if( minMoves >= L+1 )
    fout << "NOT POSSIBLE" << endl;
  else
    fout << minMoves << endl;
  fout.close();
  return 0;
}
```