/\*

\* compiled with mingw g++ 4.9.3

\*

\*

\* 匹配的时候得分为3

\* 不匹配时候得分为-1

\* 缺失时得分为-2

\*

\* 输入为两行两个字符串

\*

\* 输出为两个部分：

\* 第一部分为矩阵

\* 第二部分为最佳的匹配序列

\*

\* 注 ： 使用文本输入输出会有更好的格式体验！

\*/

#include <cstdio>

#include <vector>

#include <cstring>

#include <cstdlib>

#include <iostream>

#include <algorithm>

using namespace std;

typedef long long ll;

typedef pair<int, int> pii;

const int maxn = 10020;

const int MATCH = 3, MISMATCH = -1, OTHER = -2;

int dp[maxn][maxn], len1, len2;

char s1[maxn], s2[maxn];

string ans1, ans2;

// waterman 算法进行dp

void waterman()

{

memset(dp, 0, sizeof(dp));

for (int i = 1; i <= len1; i++)

for (int j = 1; j <= len2; j++)

{

dp[i][j] = dp[i-1][j-1] + (s1[i] == s2[j]? MATCH : MISMATCH);

dp[i][j] = max(dp[i][j], max(dp[i-1][j], dp[i][j-1]) + OTHER);

}

}

// 找到得分最高的点进行回溯答案

void getpos(int& x, int& y)

{

for (int i = 0; i < len1; i++)

for (int j = 0; j < len2; j++)

if (dp[i][j] > dp[x][y])

{

x = i;

y = j;

}

}

// 输出答案

void output()

{

int px = len1, py = len2;

ans1.clear(), ans2.clear();

getpos(px, py);

while(!(px == 0 && py == 0))

{

if (dp[px][py] == dp[px-1][py] - 2 || py == 0)

{

ans1 = '-' + ans1;

ans2 = s2[px--] + ans2;

}

else if (dp[px][py] == dp[px][py-1] - 2 || px == 0)

{

ans1 = s1[py--] + ans1;

ans2 = '-' + ans2;

}

else if (dp[px][py] == dp[px-1][py-1] + 3 || dp[px][py] == dp[px-1][py-1] - 1)

{

ans1 = s1[py--] + ans1;

ans2 = s2[px--] + ans2;

}

}

cout << ans1 << endl;

cout << ans2 << endl;

}

int main()

{

// freopen("data.in", "r", stdin);

// freopen("data.out", "w", stdout);

scanf("%s%s", s1+1, s2+1);

len1 = strlen(s1+1);

len2 = strlen(s2+1);

waterman();

printf("\t#");

for (int i = 0; i <= len2; i++)

printf("\t%c", s2[i]);

printf("\n");

for (int i = 0; i <= len1; i++)

{

printf("\t%c", s1[i]);

for (int j = 0; j <= len2; j++)

printf("\t%d%c", dp[i][j], j == len2? '\n':'\0');

}

output();

return 0;

}