#include <bits/stdc++.h>

#include<algorithm>

using namespace std;

//LCS Variations

/\*

diff utitlity ,

min. insertion/del to convert s1 to s2 (m-l deletion and n-l insertion {where l= length of LCS}),

shortest common superseq ,

longest palindromic subseq , (make another reverse of string and find lcs of both then the length of that is the longest pal subseq)

longest repeating subseq , (use same string in lcs and change it when last char match and other char should'nt be at same pos then don't increment ;go to else part and take max O(m\*n))

space optimized DP of LCS ,

printing LCS

\*/

int tLCS(string s1, string s2) //O(m\*n)

{

int m = s1.length(), n = s2.length();

int dp[m+1][n+1];

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

dp[i][0]=0;

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

dp[0][j]=0;

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

{

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

{

if(s1[i-1]==s2[j-1])

dp[i][j] = 1 + dp[i-1][j-1];

else

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

}

}

return dp[m][n];

}

int main() {

string a="ABXRTY";

string b="ABRXY";

cout<<tLCS(a,b);

return 0;

}