Given 2 strings, check if they are anagrams. An anagram is a rearrangement of the letters of one word to form another word. In other words, some permutation of string A must be same as string B.

**Input Format**

First line of input contains T - number of test cases. Its followed by T lines, each line containing 2 space separated strings.

**Constraints**

10 points  
1 <= T <= 100  
1 <= len(S) <= 103  
'a' <= S[i] <= 'z'

40 points  
1 <= T <= 100  
1 <= len(S) <= 105  
'a' <= S[i] <= 'z'

**Output Format**

For each test case, print True/False, separated by newline.

**Sample Input 0**

4

a a

b h

stop post

hi hey

**Sample Output 0**

True

False

True

False

**Explanation 0**

Self Explanatory

#include <map>

#include <vector>

#include <string>

#include <iostream>

using namespace *std*;

bool isAnagrams(*string* s1, *string* s2)

{

if (s1.*length*() != s2.*length*())

return false;

if (s1.*empty*() || s2.*empty*())

return false;

*map*<char, int> ms1;

*map*<char, int> ms2;

for (auto i = 0; i < s1.*length*(); i++)

{

ms1[s1[i]] += 1;

ms2[s2[i]] += 1;

}

return (*std*::*equal*(ms1.*begin*(), ms1.*end*(), ms2.*begin*()));

}

int main()

{

*ios*::*sync\_with\_stdio*(0);

*cin*.*tie*(0);

*cout*.*tie*(0);

int t; *cin* >> t;

while (t--)

{

*string* s1, s2; *cin* >> s1 >> s2;

if (isAnagrams(s1, s2))

*cout* << "True\n";

else

*cout* << "False\n";

}

return 0;

}