**PROBLEM NUMBER 1:**

Find the least number of characters of the first string ‘S1’ needed to be altered to make it a re-arranged word of the second string ‘S2’.

***Note:*** A word *S1* is a re-arranged word of another word *S2* if we can produce *S1* by rearranging the letters of *S2*.

**Input Format**

The first line will contain an integer ‘T’ representing the number of test cases. Each test case will contain a string having length len(S1) + len(S2) , which will be the concatenated string of both the strings ‘S1’ and ‘S2’ described above in the problem. The given string will contain only characters from ‘a’ to ‘z’.

**Output Format**

An integer corresponding to each test case is printed in a different line, i.e. the number of changes required for each test case. Print -1 if it is not possible.

**Sample Input**

6

aaaabbbb

abca

abc

mnmp

lkkl

gagbbbgg

**Sample Output**

4

1

-1

1

0

1

**Explanation**

*Test Case #01:* We have to replace all three characters from the first string to make both of strings “re-arranged”. Here, S1  = "aaaa" and S2 = "bbbb". We have to replace all ‘a’s in S1 with ‘b’s to make it a “re-arranged” word of S2.   
*Test Case #03:* It is not possible for two strings of unequal length to be ‘re-arranged words’ for each other. 

**PROBLEM NUMBER 2**:

Given a string “S”, find its largest palindromic substring.

**Input Format**

A string S

**Constraints**   
 All the characters are lower-case English letters.

**Output Format**

Print the largest possible substring of S that is a palindrome.

**Sample Input 1**

rotator

**Sample Output 1**

rotator

**Sample Input 2**

leveling

**Sample Output 2**

Level

**Sample Input 3**

hellonmadamgoodafternoon

**Sample Output 3**

madam

**PROBLEM NUMBER 3**:

Make a binary search tree. The node of the tree should contain a unique integer key.

Write the following three functions.

* A function to insert an element.
  + Void insert(int key)
* To print the tree in breadth first traversal. (Level Order)
  + Void print();
* Write an Efficient Function to Convert a Binary Tree into its Mirror Tree.
  + Void makeMirrorImage();

So the tree...

       4

      / \

     2   5

    / \

   1   3

 Is changed to...

       4

      / \

     5   2

        / \

       3   1