**Substrings**

**Time Limit: 2000/1000 MS (Java/Others)    Memory Limit: 65536/32768 K (Java/Others)  
Total Submission(s): 9134    Accepted Submission(s): 4312**

Problem Description

You are given a number of case-sensitive strings of alphabetic characters, find the largest string X, such that either X, or its inverse can be found as a substring of any of the given strings.

Input

The first line of the input file contains a single integer t (1 <= t <= 10), the number of test cases, followed by the input data for each test case. The first line of each test case contains a single integer n (1 <= n <= 100), the number of given strings, followed by n lines, each representing one string of minimum length 1 and maximum length 100. There is no extra white space before and after a string.

Output

There should be one line per test case containing the length of the largest string found.

Sample Input

2

3

ABCD

BCDFF

BRCD

2

rose

orchid

Sample Output

2

2

题意：找出一个子串x 使x或者x的逆串使者n个字符串的公共子串 输出x的最大的长度

分析：

要想找到公共子串那么只需找到最短的字符串 然后枚举所有的子串 然后在一个一个找 还好范围是100 不会超的

AC代码：

#include <stdio.h>

#include <string.h>

#include <algorithm>

using namespace std;

int main()

{

int t,n,i,j,k,MIN,f,len,MAX;

char str[105][105],s1[105],s2[105];

scanf("%d",&t); ///t组测试数据

while(t--)

{

scanf("%d",&n); ///n个字符串

MIN = 1000;

for(i = 0; i<n; i++)

{

scanf("%s",str[i]);

len = strlen(str[i]);

if(MIN>len)///找到最小串

{

MIN = len;

f = i;

}

}

len = strlen(str[f]); ///最小串的长度

int flag = 1;

MAX = 0;

for(i = 0;i<len;i++)///作为标本串子串的头

{

for(j = i;j<len;j++)///子串的尾

{

for(k = i;k<=j;k++)///复制为两个串，顺序串s1，逆序串s2

{

s1[k-i] = str[f][k]; ///s1正序 s2倒序

s2[j-k] = str[f][k];

}

s1[j-i+1] = s2[j-i+1] = '\0'; ///'\0'之前是s1串和s2串的实际长度

int l = strlen(s1);

for(k = 0;k<n;k++)///枚举所有串

{

///strstr(s,s1) 在s串中找s1串出现的位置 返回值 bool型 找到 true 否则 fal

if(!strstr(str[k],s1) && !strstr(str[k],s2))

{

flag = 0;

break;

}

}

if(l>MAX && flag)

MAX = l;

flag = 1;

}

}

printf("%d\n",MAX);

}

return 0;

}