Programming Contest Problem Set 2012

This problem set contains 9 problems (A-I) in 17 pages.

Hosted by

Faculty of Computer and Mathematical Sciences

Universiti Teknologi MARA Perak

In collaboration with

Faculty of Computer and Mathematical Sciences

Universiti Teknologi MARA Malaysia

8-9 December 2012

A	P-LOCK	
	Input	Standard Input
	Output	Standard Output
	Time Limit	8 seconds

Problem Description

RSecurity Sdn. Bhd. is a company noted in manufacturing digital security locks in Malaysia. They are designing a new product called P-Lock, as illustrated below. Prior to commercializing the product, the company wanted to carry out a final test to check the reliability of the security system embedded in the product. The security-alphabet specifications are as follows:

i. there could be a combination of ***n*** alphabets, where $2 < \mathbf{n} < 27$.
ii. each security-alphabet combination contains uppercase alphabet A to Z only. Any other characters such as numbers and punctuations should not be considered as a valid combination.
iii. each security-alphabet combination is of different alphabets only.

Input

The first line of input is **T**, *1 <= T <= 10000*, the number of test cases, followed by **T** lines; one line for each test case. Each test case contains a lock combination **L**.

Output

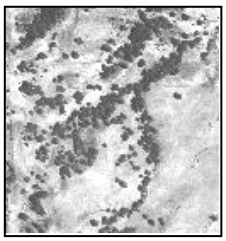
For each test case, the output contains a line in the format Case #x: **M**, where x is the case number (starting from 1). **M** is "VALID" when the given lock combination is valid, or "INVALID" if otherwise. Quotes are for clarity only.

Sample Input	Sample Output
5 TRXKR PTD R S SCQOMXNZIFRYAGPE MC9YZ	Case #1: INVALID Case #2: VALID Case #3: INVALID Case #4: VALID Case #5: INVALID

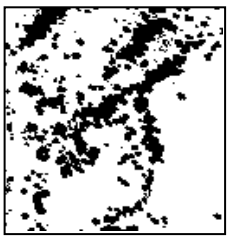
B	THE MATRIX RELOADED!	
	Input	Standard Input
	Output	Standard Output
	Time Limit	8 seconds

Problem Description

Black and white colours are common in image analysis. A gray or coloured-image will normally be converted into a black-and-white image to gain its representations. One of the representations is in a series of ones and zeros format as depicted below. The figure shows (a) original image, (b) black-and-white image, and (c) a sample of ones and zeros representation.



(a)



(b)

```
0000111000111110
0000111110000000
1110000000111111
0000011100000011
```

(c)

Input

The first line of input is **T**, *1 <= T <= 10000*, the number of test cases, followed by **T** lines. Each test case contains a series of ones and zeros of length between 101 and 65534 inclusive.

Output

For each test case, the output contains a line in the format Case #x: **M**, where x is the case number (starting from 1) and **M** is the number of ones block. A ones block is defined as a block that contains a sequence of one or more ones. **M** is -1 if the data range is invalid.

Sample Input	Sample Output
3 00000000000000000111111111100000 00111111111100000000000000000000 01111100000000000000000000000111 000000000000000000000000 11111111111111111111111111111111 11111111111111111111111100000000 00000000000000000000000000000000 00000000000000000000000000000000 00000001111111111111111111111111 11111111111111111111111111111111 11111111111111111111111111111111 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00000000000000000000000000000000 00001111111111111111111111111111 11111111111111111111111111111111 1111111100000000 11110001110000000000000001110000 11110000001110011110000000001111 1100000	Case #1: 4 Case #2: 3 Case #3: -1

C	HAIL-FAST TAXI	
	Input	Standard Input
	Output	Standard Output
	Time Limit	10 seconds

Problem Description

The *Teksi Cepat* company wants to ensure its customers do not have to wait long to hail down a taxi. There are always taxis available at the station to service a customer, but the customer may be served faster by a vacant taxi plying on a road close to the customer.

The company tracks the location of the customers and each of its vacant taxis on a *20000* by *20000* grid similar to an (X, Y) plane. The taxi station is located at the centre, i.e., at (0, 0). You are required to write a program that determines the taxi that should be directed to a customer.

Input

The first line of input contains an integer **T** *(1 ≤ T ≤ 100)*, the number of test cases. On the first line of each test case is the coordinate of the customer, followed by the coordinates of each of the vacant taxis that are currently on the road. The coordinate is a pair of integers (**X**, **Y**), where *-10000 ≤ X, Y ≤ 10000*, separated by a single space. Each coordinate will be on a separate line. (10001, 10001) will signify the end of each test case.

Output

The output comprise of one line for each test case set. The line begins with the prefix "Case #x:" where x represents the case number (starting from one and incrementing at each new test case), followed by a single space, and then the coordinate of taxi to be directed to the customer. The X-coordinate and the Y-coordinate are separated by a single space. If there is more than one solution, only one must be selected – in such situations, the selection priority order is the taxi

located at the top, followed by the taxi located on the right hand side on the grid.

Sample Input	Sample Output
2 5 5 3 -2 -5 -3 7 8 10001 10001 -3 8 10 6 1 2 10001 10001	Case 1: 7 8 Case 2: 1 2

D COUNTING SUBSEQUENCES

Input	Standard Input
Output	Standard Output
Time Limit	8 seconds

Problem Description

We can define that a subsequence is a sequence that can be obtained by deleting one or more elements from another sequence while maintaining the order of the elements.

From this definition, we can conclude that a string of length N has 2^N number of subsequence. For example, "abad" has 16 subsequences: { "", "a", "b", "a", "d", "ab", "aa", "ad", "ba", "bd", "ad", "aba", "abd", "aad", "bad", "abad" }. Note that the same alphabet in different positions of the string is treated as different subsequences; thus "a" and "ad" appears two times in the subsequence list. Also remember to include the NULL element "" in the subsequence list.

Suppose **P** is a multiset containing all subsequences of string ***p,*** and **Q** is a multiset containing all subsequences of string ***q***. Your task is given two strings **p** and **q**, compute the number of subsequences **a** and **b** such that $a = b, a \in P$ *and* $b \in Q$. In other words, you have to count the number of common subsequences of the two given strings.

For example, let's consider another string "bda" which has 8 subsequences: { "", "b", "d", "a", "bd", "ba", "da", "bda" }. The subsequences of "bda" which equal to subsequences of the string "abad" are { "", "b", "d", "a", "a", "bd", "ba" }, which contains 7 elements. Notice that there are two "a" as there are two occurrences of "a" in the subsequence of "abad" and one occurrence of "a" in subsequence of "bda".

Input

The first line of input is ***T***, 1 <= *T* <= 10000, the number of test cases. The first line of each test-

case contains a string **p** while the second line contains a string **q**. Each string consists of lowercase alphabet (a-z) only and has length between 1 and 1,000 inclusive.

Output

For each test case, the output contains a line in the format `Case #x: Y`, where `x` is the case number (starting from 1) followed by a single space, followed by **Y**, the number of common subsequences of strings **p** and **q** for that case. As the result could be very large, we suggest you modulo **Y** with 1,000,000,003.

Sample Input	Sample Output
4 abad bda acm icpc programming contest vwxyz vxzw	Case #1: 7 Case #2: 3 Case #3: 4 Case #4: 10

E	DASH PATTERN	
	Input	Standard Input
	Output	Standard Output
	Time Limit	8 seconds

Problem Description

Dash lines can be represented using dash patterns. A dash pattern specifies the number of dash (`'_'`) units to be drawn followed by the number of space (`' '`) units to be skipped. For example a dash line in which both the dash and space are 4 units long is represented by a 2-dash pattern: `4 4`. The dash line of length 30 for this pattern is illustrated as follows. *Note:* The digits below the line are for clarity purposes only.

```
____ ____ ____ ____
123412341234123412341234123412
```

To obtain a more complex dash line, we can use a dash pattern that consists of 3 or more digits. For example, a dash line consists of 8 dashes, 4 spaces, 2 dashes and 4 spaces of length 30 as illustrated below. The dash line is represented by a 4-dash pattern: `8 4 2 4`.

```
________    __  ________
123456781234121234123456781234
```

Input

The first line of input is **T**, 1 <= *T* <= 10000, the number of test cases, followed by **T** lines. Each test case starts with **L**, the length of the line to be drawn, where 1 < *L* < 100, followed by an **X**-dash pattern where 1 < *X* < 10. The dash patterns are represented by digits between 1and 9 inclusive. The numbers in the test case are separated by a single space.

Output

For each test case, the output contains a line in the format `Case #x:` where `x` is the case number (starting from 1) followed by a single space, followed by a dash line of length **L**. Each dash line will starts and ends with the character `'|'`.

Sample Input	Sample Output
4 30 4 4 30 8 4 2 4 10 3 3 10 2 1	Case #1: |____ ____ ____ ____ | Case #2: |_____ __ _____ | Case #3: |___ ___ | Case #4: |__ __ __ _|

F	LEGOLAND TICKETS	
	Input	Standard Input
	Output	Standard Output
	Time Limit	8 seconds

Problem Description

LEGOLAND Malaysia opens its 76 acres of adventure on 15th September 2012. The park is packed full of family fun with more than 40 rides, shows and attractions. There are three pricing categories for the LEGOLAND tickets.

Ticket	Reg. price
1-Day Ticket Adult (age 12-59)	RM 140
1-Day Ticket Child (age 3-11)	RM 110
1-Day Ticket Senior (age 60+)	RM 110

In addition to the tickets, LEGOLAND partners with Munchy, Dutchlady and KFC to giveaway vouchers. Each voucher is worth RM110. A voucher can be exchanged for a free 1-day child ticket with the purchase of a 1-day adult ticket to LEGOLAND at the same time. You are required to write a program to calculate the total price of the tickets purchased by a customer based on the age of the ticket holders and the number of vouchers exchanged.

Input

The first line of input is **T** which is the number of cases, $1 < T < 1000$. This is followed by the test cases, one line for each test case. Each test case begins with an integer **N,** $1 < N < 100$, which represents the number of ticket holders, followed by **N** ticket holders' ages, and is ended with an integer that represents the number of vouchers exchanged.

Output

For each test case, the output contains a line in the format Case #x: where x is the case number (starting from 1) followed by a single space, and then the total price of the tickets. The total price of the tickets is formatted as follows: `RM<integer>.00`.

Sample Input	Sample Output
2 4 34 2 32 4 6 9 8 10 12 14 16 18 19 49 47 1	Case #1: RM280.00 Case #2: RM1090.00

G	EVEN (OR ODD) WINS!	
	Input	Standard Input
	Output	Standard Output
	Time Limit	8 seconds

Problem Description

Given an integer, determine if the cube of the integer consists of even digits only or odd digits only or a mixture of even and odd digits.

Input

The first line of input is **T**, *1 <= T <= 10000*, the number of test cases, followed by **T** lines. Each test case consist of an integer **N**, *0 <= N <= 1000*.

Output

For each test case, the output contains a line in the format Case #x: where x is the case number (starting from 1) followed by a single space. If the cube of **N** consists of even digits only, output "EVEN". Output "ODD" if the cube of **N** consists of odd digits only, otherwise output "MIXED". Quotes are for clarity only.

Sample Input	Sample Output
3 2 3 15	Case #1: EVEN Case #2: MIXED Case #3: ODD


PROMED
@CS'12

H STRUCTURE OF WORD

Input	Standard Input
Output	Standard Output
Time Limit	3 seconds

Problem Description

Creating a random name generator is not an easy task. There are so many factors to be considered like the origin of the name, the length etc. We can also consider the structure of the words used, which is based on the pattern of vowels and consonants. For example, the structure of the word 'GREEN' is *consonant – consonant – vowel – vowel – consonant*. Given a pair of words generated by a random name generator, determine if the words have the same structure or not.

Input

The first line of the input is a positive integer **T**, $1 < T < 100$ indicating the number of test cases. Each testcase consists of a line of two words, separated by a single space. Each word has **N** alphabets where $1 < N < 256$.

Output

For each test case, the output contains a line in the format Case #x: where x is the case number (starting from 1), followed by the result of comparing the structure of the word pairs, i.e., either they are "*same"* or "*different"*.

Sample Input	Sample Output
3 moon cake silver duster sunday blue	Case #1: different Case #2: same Case #3: different

I INSEPARABLE LUQ & RUQ

Input	Standard Input
Output	Standard Output
Time Limit	8 seconds

Problem Description

Two twins, Luq and Ruq, both managed to get scholarships to further their studies in Japan. However, the scholarship regulations disallowed siblings to study at the same university. Siblings are also disallowed to study at any university in the same state. Nevertheless, Luq and Ruq plan to stay as near as possible without breaking these regulations. You are to help them identity these two nearest universities.

Input

The first line of input is **T**, *1 <= T <= 10000*, the number of test cases. The first line of each test case is the number of universities **N**, *1 <= N <= 100000*. The next **N** lines each contain the x-coordinate and the y-coordinate of the university, (*0 <= x, y <= 100000*) followed by the state code **S**. Assume the university code is in ascending order based on the row number of each test case starting from 1 and up to 100000. The state codes are all single upper-case alphabets. All

input are separated by a single space.

Output

For each test case, the output contains a line in the format Case #x: where x is the case number (starting from 1) followed by a single space. If there is a solution, this is followed by the first chosen university code and its state code, separated by a single space, and followed by the second chosen university code and its state code, again separated by a single space. These two universities must be sorted in ascending order of their university codes. If there is no solution, print "NO SOLUTION".

Sample Input	Sample Output
2 3 4 7 A 49 99 B 1 1 C 4 9 98 B 111 0 B 101 345 B 10 4 B	Case #1: 1 A 3 C Case #2: NO SOLUTION

END OF PROBLEMS