Problem 1. Arithmetic Means of Two N×M Dimensional Arrays 兩個 N×M 維陣列的算術平均

(Time Limit: 3 seconds)

問題描述：

Write a program to take the arithmetic means of each pair of values in two N × M dimensional arrays. These calculated means will be outputted into a newly generated N × M dimensional array.

撰寫一個程式，計算兩個 N×M 維陣列中的每對數值的算術平均，計算出的平均數將輸出到另一個新的 N×M 維陣列中。

輸入說明:

The 1st line contains two integers, N M (1≤N,M≤3). It represents that input data is NxM array.

The following NxM real numbers is array A.

The following NxM real numbers is array B.

For example:

N M
A[0, 0]
A[0, 1]
A[0, 2]
…
B[0, 0]
B[0, 1]
B[0, 2]
…

第一行包含兩個整數，N M（1≤N，M≤3）。 它表示輸入數據是 NxM 陣列。

接著的 NxM 數字是陣列 A.

接著的 NxM 數字是陣列 B.

例如：

N M
A[0, 0]
A[0, 1]
A[0, 2]
...
B[0, 0]
B[0, 1]
B[0, 2]
...

輸出說明:

A new NxM array whose elements are arithmetic means of array A and array B.
Each line represents a row, and output format for elements is [content].
There is no space between [content] and [content].
There is a blank line after each test case.
※ round to one decimal places

一個新的 NxM 陣列，其元素是陣列 A 和陣列 B 的算術平均數。
每行輸出代表陣列的一列，元素的輸出格式為[content]。
[content]和[content]之間沒有空格。
每個測試用例後都有一個空行。
※ 舍入到小數點後一位

範例:

Sample Input:	Sample Output:
2 2 2 3 7 1 8 7 1 9	[5.0][5.0] [4.0][5.0]
3 2 6.5 3 1 2 4 1 1.5 7 9 8 6 7	[4.0][5.0] [5.0][5.0] [5.0][4.0]

Problem 2. Caesar Shift 字母往前移

(Time Limit: 2 seconds)

問題描述：

George and Mary make a lovely couple. They have a habit exchanging diaries to kill time. In order to protect the content from being peeked by classmates or being confiscated by teachers, they use a way that shifts characters a certain number of positions in the alphabet and this certain number is written in the next line of the content. Mary feels that it is not convenient to translate diary content into an "encrypted" format every time it is done. Please help Mary write a program that can translate diary content into an "encrypted" form. The result of the encryption does not affect the case of the original letter, and the digital part is processed in the same way. The symbols and special characters and Chinese are not processed.

志明跟春嬌是班上的一對情侶，他們有寫交換日記來打發時間的習慣，為了防止他們寫的內容被幫忙傳的同學偷看，或者是不小心被老師沒收，而曝光了裡面寫的東西，他們想到了一個辦法，就是把內容的所有字母都往後數幾次的字母替代，而往後數幾次的數目就寫在內容的下一行。但是，問題來了，春嬌覺得每次寫完都要再數來數去的轉化成「加密」格式，實在是太麻煩了。但又礙於不想被輕易的看到內容，於是她拜託你寫個程式幫忙她可以直接把寫好的內容轉化成「加密」的型態。加密結果不會影響原字母的大小寫，且數字部分亦作相同處理，但不處理符號及特殊字元及中文。

輸入說明:

The first line of the input is the content. It is not greater than 100 characters.

The second line of the input is the number of position by which each character is shifted.

第一行為想輸入的內容，不超過 100 個字元。

第二行為打完你想輸入的內容之後，換行輸入你想要往後替代的數目。

輸出說明:

Output the sentence that is translated and add a “newline” in the end of output.

印出轉換後的句子，最後必須有換行字元。

範例:

Sample Input:	**Sample Output:**
How are you? 123 2	Jqy ctg aqw? 345

Problem 3. Number Puzzle

數字猜謎

(Time Limit: 3 seconds)

問題描述 (Problem Description):

The number puzzle is a popular game for enhancing the children' s arithmetic skills. Here we will introduce a number addition puzzle. In the puzzle, we use an alphabet (A~G) to represent a number digit (0~ 9), the digits of different alphabets are distinct each other. A puzzle contains an equation $E_1 + E_2 = E_3$ with three numbers E_1, E_2, and E_3, where E_1 and E_2 are 2-digit numbers and E_3 is a 3-digit number. The goal of the puzzle is to recover the digits of the numbers. For example, one answer of the following puzzle "AB + BC = EED" is "A = 8", "B = 3", "C = 2", "D = 5", and "E = 1".

數字猜謎是一個加強小孩算術技能的遊戲。這裡將介紹加法的猜謎。在一個猜謎中，我們以英文字母(A~G)分別代表(0~9)某個數字。一個猜謎包含一個等式：$E_1 + E_2 = E_3$ 其中 E_1 與 E_2 為兩位數的數字，E_3 三位數的數字。此猜謎的目的為把利用A~G所表達的數字還原成由0~9所表示的數字。例如："AB + BC = EED"，"A = 8"，"B = 3"，"C = 2"，"D = 5"，及 "E = 1".

$$\begin{array}{r} AB \\ +\ BC \\ \hline EED \end{array}$$

For any given puzzle, there may have multiple answers or no answer. Please write a program to verify whether a given puzzle has answers or not.

對於每個給予的猜謎，可能會有多個解答或是沒有解答。請寫一支能夠用來判斷給予的猜謎有沒有答案的程式。

輸入說明 (Input Format) :

The input consists of several test cases. For each test case, there have three lines of alphabet-numbers. The first and the second lines consist of 2 alphabets and the third line consists of 3 alphabets. Notice that different alphabets represent distinct number digits.

輸入包含幾個測式案例。每個測試案例有三列用來表示字母數字。第一與第二列由兩個字母組成，第三列由三個字母組成。注意不同字母代表不同的數。

輸入說明 (Output Format) :

The output contains one line for each test case. If there exist answers for the puzzle, then output “YES”, otherwise output “NO”.

每個測試案例輸出只有一列。如果測驗案例有答案，則輸出”YES”, 否則輸出”NO”。

Please add a “newline” in the end of the output.

在輸出結果的最後請加一個“newline”。

範例 (Example) :

Sample Input:	Sample Output:
AB BC EED AA BB ABA	YES NO

Problem 4. Merge Sort

合併排序

(Time Limit: 2 seconds)

問題描述：

Merge sort is a sorting algorithm. It divides numbers into two groups first, then sorts the numbers in these two groups respectively, and finally merges them. For example, A and B represent two groups of sorted numbers. A is (2, 4, 5) and B is (1, 6). Two pointers which point to the first number of A and B respectively can be used when doing merge process. Then output the smaller number pointed by these two pointers. In this case, the first output number is 1. The pointer for B points to next number. Output the smaller number pointed by these two pointers again. Now the second output number is 2. The pointer for A points to next number. Please write a program that can sort two groups of numbers according to merge sort algorithm described above.

合併排序是一種將一組數字拆成兩組，待這兩組數字分別排序後進行合併的動作。例如 A 與 B 分別表示兩組已經排序好的數字， A 為 (2, 4, 5) 且 B 為 (1, 6) 。合併的方式可以透過兩個指標，分別指向 A 與 B 的第一個數字，接著輸出這兩個指標所指較小的數字，因此第一個輸出數字為 1 。緊接移動 B 的指標至下一個數字，再輸出這兩個指標所指較小的數字，因此第二個輸出數字為 2 。接著移動 A 的指標至下一個數字，以此類推。請寫一個程式，完成兩組數字的合併排序。

輸入說明：

For each test case, the input should consist of two lines with a sorted number list. For example, enter N numbers in the first line and M numbers in the second line, where N, M ≤ 10.

If numbers in the first line of a test case are all 0, then program ends;

每一測試資料需要輸入兩行以單一空白區隔並由小至大排列的整數數字，其中第一行輸入 N 個數字，第二行輸入 M 個數字。 N, M ≤ 10。

如果測試資料的第一行全部輸入數字 0 則結束程式執行。

輸出說明：

Output sorted numbers separated by one space and add a “newline” in the end of output.

輸出排序後的數字，數字間以一個空格隔開，最後必須有換行字元。

範例：

Sample Input:	Sample Output:
2 5 6 8 12 1 4 15 1 8 9 11 2 4 5 10 13 0 0 0 0 0	1 2 4 5 6 8 12 15 1 2 4 5 8 9 10 11 13

Problem 5. Lowest Non-zero Digit of Factorial

階層最低非零數字

(Time Limit: 2 seconds)

問題描述：

The factorial of *n*, denoted by *n*!, is *n* × (*n*-1) × ... × 2 × 1. When we write *n*! in 7-ary numeral system, there are trailing zeroes for *n* ≥ 7. For instances, we have $7! = 10_7 \times 6_7 \times ... \times 2_7 \times 1_7 = 20460_7$ and $14! = 20_7 \times 16_7 \times ... \times 2_7 \times 1_7 = 6204234151200_7$. We are interested in the lowest non-zero 7-ary digit of *n*!, so you don't have to store every digit of *n*! . This problem can be easily solved when *n* is small by the following C function **lnzsdf.**

n 階層（以 *n*!表示）的計算方式為 *n* × (*n*-1) × ... × 2 × 1。如果我們利用七進位數字系統計算 *n*!，當 *n* ≥ 7 時，最後計算的結果的尾數有可能包含數個零。例如，7!的七進位計算式為 $7! = 10_7 \times 6_7 \times ... \times 2_7 \times 1_7 = 20460_7$（尾數一個零），而 14!的七進位計算式為 $14! = 20_7 \times 16_7 \times ... \times 2_7 \times 1_7 = 6204234151200_7$（尾數二個零）。在這個計算範例中，我們對計算結果的數字裡哪個是位數最低的非零數字感到興趣。例如，20460 是 6，6204234151200 是 2。這個問題可以利用底下的 C 程式函數 **lnzsdf** 得知：

```
int lnzsdf(int x){
    int i, j, ret = 1;
    for(i = 1; i <= x; i++){
        for(j = i; j % 7 == 0; j /= 7);
            ret = (ret * (j % 7)) % 7;
    }
    return ret;
}
```

However, this is not fast enough to solve large *n*. Please design a new algorithm to solve this problem.

然而，當 *n* 值很大時，這個方法並不夠快。請設計新的演算法解決這個問題。

輸入說明：

The first line of the input contains a integer $T \leq 20$ which is the number of test cases. Each test case consists of one number n where $1 \leq n \leq 2000000000$.

第一行輸入包含一個整數 T（$T \leq 20$）表示測資的筆數。第二行開始為各筆測資。每筆測資包含一個數字 n，其中 $1 \leq n \leq 2000000000$。

輸出說明：

The output contains one line for each test case. Each line contains the lowest non-zero 7-ary digit of $n!$ which equals **lnzsdf(n) .**

Please add a “newline” in the end of the output.

每筆測資請以一行輸出計算結果，即 **lnzsdf(n)**的答案。最後一行輸出結尾亦須換行。

範例：

Sample Input:	**Sample Output:**
4 7 14 100 2000000000	6 2 4 1

Problem 6. Making Change

零錢兌換

(Time Limit: 2 seconds)

問題描述：

A store owner frequently encounters such a problem that a customer requests a less number of coins to make the change. Let us help the store owner to solve this problem. Suppose that the owner has a variety of valued coins and has an infinite supply of each valued coin. We need to find the minimum number of coins to make the change. For example, we want to make the change for 5321 dollars, and the valued coins the customer expects are 1000, 200, 20, and 1. The store owner makes such a change of 1000*5+200*1+20*6+1*1, which involves 13 coins.

市場的老闆在進行金錢交易的時候經常會遇到顧客要求找的零錢總數愈少愈好，因此我們要幫助老闆解決此問題。假設老闆擁有各種金錢面額且每種面額總數無上限，因此老闆可以依據每位顧客所要求的兌換面額進行最少零錢總數的兌換。假設要找的錢為 5321 且顧客要求的面額為 1000 、200 、 20 和 1 ，則老闆必須找給顧客 1000*5+200*1+20*6+1*1 共 13 個零錢總數。

輸入說明:

The first input line contains a number which indicates the number of test cases. The test data start from the second line. Each line of the test data includes several numbers of which two numbers are separated by a space. Among them, the last number (1~100000) is the value of making the change, and the remaining numbers are valued coins (each having a value between 1 and 100000) in descending order. The last valued coin must be 1.

輸入的第一行是代表測試資料有幾組，接著第二行開始為測試資料，每行測試資料的最後一個數字代表所要找的金錢(1~100000)，前面的數字則代表顧客所要求的兌換面額(1~100000)，面額須由大至小輸入且最後一個面額一定要為 1。數字間均有空白間隔。

輸出說明

For each test case, show the minimum number of coins to make the change in the first line; and in the following lines, show each valued coin in descending order and the corresponding number of coins. Use a space to separate these two numbers. Please add a newline character at the end of the output.

依序輸出每組測試資料的結果，且每組輸出資料的第一行代表最少的零錢數，之後依面額大小順序依序輸出每種面額以及其所兌換的個數，面額和個數之間必須有空白間隔。輸出資料最後一行結尾須加上一個換行字元。

範例:

Sample Input:	Sample Output:
2 1000 600 11 2 1 98654 600 22 3 1 54321	108 1000 98 600 1 11 4 2 5 1 0 109 600 90 22 14 3 4 1 1