#Problem 1: Complex Coin Change

Description:
You will be given N coins and an amount K. You can use each coin an infinite number of times. You have to print the minimum number of coins needed to make the amount K.

In the first line you will be given N and K. In the following line, you will be given N values denoting N coins.

Limits
1<=N<=30
1<=K<=10000

Test Cases:

Input	Output
3 7 1 2 3	3
11 30 1 13 20 23 36 37 38 51 61 94 97	6
11 73 1 13 20 23 36 37 38 51 61 94 97	2
11 500 1 13 20 23 36 37 38 51 61 94 97	6

#Problem 2: Complex Coin Change with Coin Print

Description:
You will be given N coins and an amount K. You can use each coin an infinite number of times. You have to make the amount K using the minimum number of coins. You also have to print which coins have been taken with their usage count in ascending order over the coin's value aka smaller valued coin will be printed first.

In the first line, you will be given N and K. In the following line, you will be given N values denoting N coins.
For each input, print the coins with their usage count in ascending order. If there exists multiple solutions print any of them. Print each coin's output in separate lines. Look at the input output section for more clarification.

Limits

1<=N<=30
1<=K<=10000

Test Cases:

Input	Output
3 7 1 2 3	1 1 3 2
11 30 1 13 20 23 36 37 38 51 61 94 97	1 4 13 2
11 73 1 13 20 23 36 37 38 51 61 94 97	36 1 37 1
11 500 1 13 20 23 36 37 38 51 61 94 97	51 1 61 1 97 4

#Problem 3: 0/1 Knapsack

Description:
You will be given N items and a knapsack weight W. Each item N[i] has two elements, its total positive benefit b[i] and its total weight w[i]. You have to choose the elements in such a way that you can maximize your total benefit not exceeding W. You can not take a fractional amount of weight for an item.

In the first line, you will be given N and W. In the first following line you will be given N values, each element denotes the total positive benefit where i^{th} value is for the i^{th} element. In the second following line, you will again be given N values, each denoting the weights where i^{th} value denotes the total weight for i^{th} element.

For each input, you have to output the maximum positive benefit you can achieve not exceeding the knapsack weight.

Limits
1<=N<=30
1<=W<=1000

Test Cases:

Input	Output

4 10 2 3 5 7 2 3 5 7	10
3 30 25 10 30 10 20 30	35
10 37 86 87 24 87 15 5 87 96 76 16 14 31 6 56 43 69 33 39 50 24	111
10 65 86 87 24 87 15 5 87 96 76 16 14 31 6 56 43 69 33 39 50 24	206
10 29 86 87 24 87 15 5 87 96 76 16 14 31 6 56 43 69 33 39 50 24	110

#Problem 4: LCS and Path Print

Description:
Given two strings M and N. You need to print the longest common subsequence (LCS) length found between M and N. You also need to print a lcs which exists between M and N.

In the first line, you will be given M and in the second line you will be given N. M and N will contain only digits or English alphabets.

As output, in the first line print the lcs length. In the second line, print a lcs which exists between M and N. If there exists multiple solutions print any of them. Look at the input output section for more clarification.

Limits
1<=|M|<=40
1<=|N|<=40
|M|, |N| denote the length of string M and N respectively.

Test Cases:

Input	Output
AAAB AAAB	4 AAAB
ACAAB	5

ACDAAB	ACAAB
ECBEBECABB BACACABECAEBEBA	7 BBECABB
EBEEA AAECC	1 A
EECECABBBC BBCCBEACAEEB	5 CECAB
CECCBCCECAACABEAABBCCBCECCBCEECECAE ABEACCBBACCEAEEBCABECAAAAAE	16 ABEACCBBCEECECAE