Make Change

Program Name: Change.java **Input File: change.dat**

Perhaps you have worked at one of the many fast-food restaurants located in every town? Many customers find it entertaining to see if the cashier can calculate the correct change. Confusing the cashier is offered as evidence of the collapse of modern society (or at least, our educational system), hence the entertainment value of the game. Computer controlled cash registers have taken much of the fun out of the game, but there are modern variants, typically involving giving an odd amount of cash – for example, $15.13 for a $9.63 purchase.

The *change-making* problem, also known as the minimum coin change problem, enters the picture after the correct change has been calculated, and answers the question: how many of each coin denomination to give?

You may have written a program to calculate the answer in your CS1 class using a greedy algorithm – but that solution assumes denominations like those used in the U.S. Using the naïve (greedy) solution, you find the minimum number of coins by repeatedly subtracting the largest denomination coin from the change due while the result is positive, then repeating with the next largest coin, until the coins sum to the total change due. For example, to make $0.68 change, subtract 1 half-dollar, leaving 18¢. A quarter would be too much, so subtract 1 dime, leaving 8¢. Finally, use a nickel, then 3 pennies. We use 6 coins to make the correct change.

If our selection of coins contains a different set of denominations, the greedy solution may not work. Given a set of denominations, solve for the fewest number of coins using dynamic programming.

Input

The first line of the input file will contain a single integer, s, $1 \leq s \leq 50$, giving the number of datasets to follow.

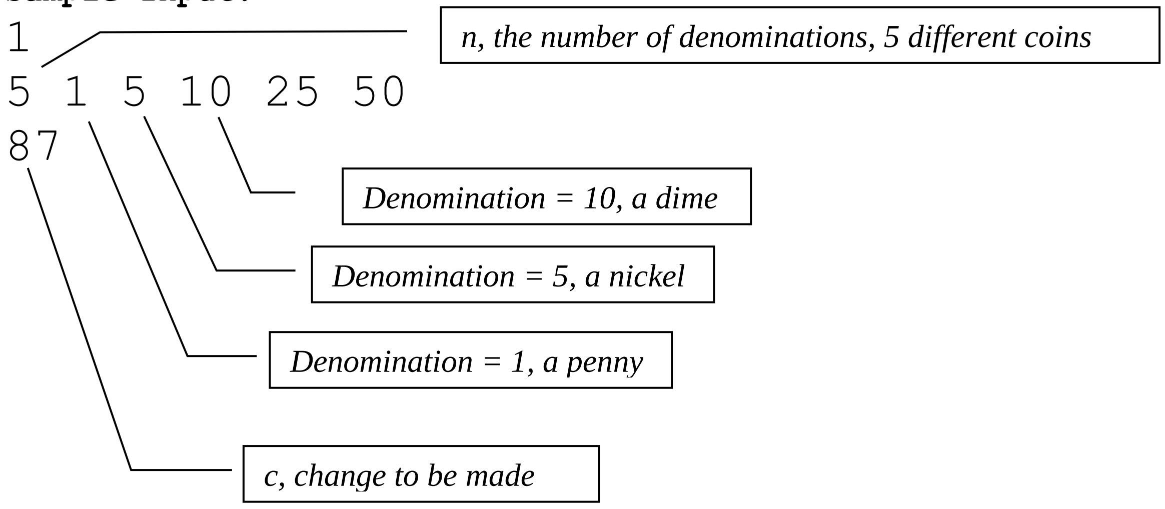
Each dataset will be in the following format:

The first integer of each dataset (line) will contain an integer n, $1 \leq n \leq 50$, the number of denominations in the currency, followed by n integers, the value of each coin. These values are ordered arbitrarily. The next line contains a single integer c, the value of the change to be made.

Output

For each dataset, output a single line of integers: first, the minimum number of coins needed to make change, followed by a list of how many coins are needed for each denomination, with denominations in the order given in the input.

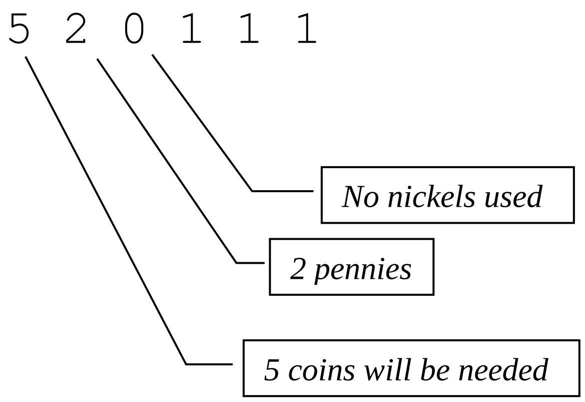
Sample Input:



```
1
5 1 5 10 25 50
87
```

n, the number of denominations, 5 different coins
Denomination = 10, a dime
Denomination = 5, a nickel
Denomination = 1, a penny
c, change to be made

Sample Output:



```
5 2 0 1 1 1
```