Making Change

Templonia is a very strange place. On one hand, there are beautiful temples, great ancient cities and many nice people. On the other hand, the Templonians have acquired some very peculiar customs. The one which causes the biggest problem for Lea is that Templonian merchants always want their money in as few coins or notes as possible. Even worse, there are many different coins and notes for the Templonian Column, and not every merchant takes all values. Surely, the Templonian people are accustomed to this behaviour and are very good in calculating these things. But as a foreign tourist, Lea has some difficulties to say the least. A tolerant person as she is, she does not want to anger the local merchants and tries to adopt Templonian customs. Whenever she has an unusually hard problem at finding the right coins, she takes out her phone and calls you. Can you please help her not get beaten up by angry merchants?

Input

The first line of the input contains an integer t. t test cases follow, each of them separated by a blank line.

Each test case starts with a single line containing two integers n and c. n is the number of coin and note values and c is the amount of money that must be spent. The next line consists of n distinct integers $v_1, \ldots, v_n$ describing the coin/note values in increasing order. You may assume that a coin of value 1 is always available, and that Lea has as many of all the coins/notes as she needs.

Output

For each test case, output one line containing "Case #i: x" where i is its number, starting at 1, and x is a space-separated sequence of n integers $a_1, \ldots, a_n$, where a_i means that the optimal solution uses exactly a_i coins/notes of value v_i.

Constraints

- $1 \leq t \leq 20$
- $1 \leq n \leq 100$
- $1 \leq c \leq 10^5$
- $1 \leq v_i \leq 10^4$ for all $1 \leq i \leq n$
- $1 \leq a_i \leq c$ for all $1 \leq i \leq n$

Sample Input 1	Sample Output 1
2 6 29 1 2 4 5 6 8 4 43 1 6 7 13	Case #1: 0 0 0 1 0 3 Case #2: 0 5 0 1

Sample Input 2

```
12
4 780
1 6 35 97

9 827
1 15 17 26 64 70 79 88 98

5 598
1 21 45 64 68

2 756
1 55

5 507
1 23 33 35 58

6 789
1 9 70 73 79 98

7 418
1 15 19 64 74 87 88

10 481
1 13 16 27 38 44 51 59 75 100

7 979
1 9 11 49 68 76 100

5 932
1 57 80 81 89

10 612
1 6 28 30 36 67 69 74 96 99

3 306
1 18 19
```

Sample Output 2

```
Case #1: 4 0 0 8
Case #2: 1 0 0 0 0 2 0 0 7
Case #3: 0 1 1 3 5
Case #4: 41 13
Case #5: 0 0 2 1 7
Case #6: 1 0 2 0 2 5
Case #7: 0 0 1 1 1 3 0
Case #8: 0 1 0 0 0 0 0 2 2 2
Case #9: 0 0 1 0 1 0 9
Case #10: 1 1 0 2 8
Case #11: 0 0 0 1 0 0 0 0 4 2
Case #12: 0 17 0
```