code jam
cout << "hello, world!" << endl;

Round 1C 2015

Ask a question

View my submissions

- Submissions

Brattleship

11pt	Correct **3434/4108 users** correct (84%)
22pt	Correct **2388/3344 users** correct (71%)

Typewriter Monkey

11pt	Correct **1651/2255 users** correct (73%)
22pt	Not attempted **575/721 users** correct (80%)

Less Money, More Problems

11pt	1 incorrect attempt **1614/2209 users** correct (73%)
23pt	Not attempted **416/737 users** correct (56%)

- Top Scores

Klockan	100
Vitaliy	100
linguo	100
y0105w49	100
Endagorion	100
wata	100
alexey.zayakin	100
apiad	100
Baklazan	100
tkociumaka	100

Problem C. Less Money, More Problems

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 11 points	Solve C-small
Large input 23 points	Solve C-large

Problem

Up until today, the nation you live in has used **D** different positive integer denominations of coin for all transactions. Today, the queen got angry when a subject tried to pay his taxes with a giant sack of low-valued coins, and she just decreed that no more than **C** coins of any one denomination may be used in any one purchase. For instance, if **C** = 2 and the existing denominations are 1 and 5, it is possible to buy something of value 11 by using two 5s and one 1, or something of value 12 by using two 5s and two 1s, but it is impossible to buy something of value 9 or 17.

You cannot directly challenge the queen's decree, but you happen to be in charge of the mint, and you *can* issue new denominations of coin. You want to make it possible for *any* item of positive value at most **V** to be purchased under the queen's new rules. (Note that this may not necessarily have been possible before the queen's decree.) Moreover, you want to introduce as few new denominations as possible, and your final combined set of pre-existing and new denominations may not have any repeats.

What is the smallest number of new denominations required?

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each consists of one line with three space-separated values **C**, **D**, and **V**, followed by another line with **D** distinct space-separated values representing the preexisting denominations, in ascending order.

Output

For each test case, output one line containing "Case #x: y", where x is the test case number (starting from 1) and y is the minimum number of new denominations required, as described above.

Limits

1 ≤ **T** ≤ 100.
Each existing denomination ≤ **V**.

Small dataset

C = 1.
1 ≤ **D** ≤ 5.
1 ≤ **V** ≤ 30.

Large dataset

$1 \le \mathbf{C} \le 100$.
$1 \le \mathbf{D} \le 100$.
$1 \le \mathbf{V} \le 10^9$.

Sample

Input	Output
4 1 2 3 1 2 1 3 6 1 2 5 2 1 3 3 1 6 100 1 5 10 25 50 100	Case #1: 0 Case #2: 1 Case #3: 1 Case #4: 3

Note that Cases #3 and #4 are not within the limits for the Small dataset.

In Case #1, it is already possible to make all the required values (1, 2, and 3) using at most one copy of each of the existing denominations.

In Case #2, it suffices to add a denomination of either 3 or 4 -- whichever you choose, only one new denomination is required.

In Case #3, the optimal solution is to add a denomination of 1.