Problem A - Smart Customer

Being a smart customer, you need figure out the cheapest way to buy things. However, the cheapest way may not be so straight forward even the price list is clearly given. As a programmer with adequate programming and algorithm knowledge, you are asked to write a program to find out the lowest price to buy a specific amount of goods, with the price list given.

Input and Output

The input consists of a number of test cases, each concerning a specific item. The first line of each case gives the unit price of buying **an item**, then a **non-negative integer** M (≤ 20). This is followed by M lines each containing two numbers N and P ($1 < N \leq 100$), which means that you can buy N such items for $P. Finally there is a line containing a list **(at most 100)** of **positive integers** K (≤ 100), each separated by a single space; for each of them your program should print the lowest price you need to get **exactly** K items. Note that all prices P in the input/output are floating-point numbers in exactly 2 decimal places, with $0 < P < 1000$.

Sample Input

```
22.00 2
2 22.00
4 60.00
2 4
25.00 2
2 48.00
2 46.00
2
22.00 2
2 22.00
4 40.00
1 2 3 4
```

Sample Output

```
Case 1:
Buy 2 for $22.00
Buy 4 for $44.00
Case 2:
Buy 2 for $46.00
Case 3:
Buy 1 for $22.00
Buy 2 for $22.00
Buy 3 for $44.00
Buy 4 for $40.00
```

Tips:

To make your life easier, the problem-setter would like to remind you about the following tips:

1. Being a smart student, you should probably know by now that this problem is asking you to find an **optimal solution**.
2. When you see floating point numbers in input/output, you should know that carefulness of handling those numbers is important.
3. Please be careful about the terms like "**non-negative integer**" and "**positive integers**", problem-setter is quite sure that you know what they mean.
4. Finally, don't ever underestimating the limits of test cases. When the problem-setter sets those limits, he will probably test them.