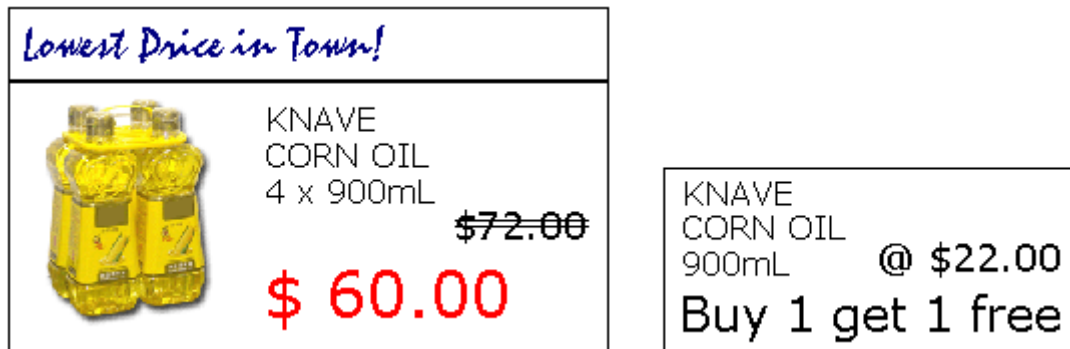


## 10980 Lowest Price in Town

It is sometimes tricky to figure out the *cheapest* way to buy things, even in the supermarket where the price of all goods are listed clearly. Just consider what I saw last Saturday about the price of cooking oil: (notice the difference in the sizes of the two price tags)



Having a sharp mind (a consequence of regularly taking part in online programming contests), you should have no problem in seeing that the ‘buy-1-get-1-free’ scheme is preferable. But what about your Mum? It is your responsibility as her son/daughter to **write her a program** that computes the lowest price to buy things in the supermarket, thus helps her to save money.

### Input

The input consists of more than a hundred test cases, each concerning a different item. The first line of each case gives the unit price of buying an item, then a non-negative integer  $M$  ( $\leq 20$ ). 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 of positive integers  $K$  ( $\leq 100$ ).

### Output

For each of them your program should print the lowest price you need to get  $K$  items. Note that you do **not** have to buy *exactly*  $K$  items; you may consider buying more than  $K$  items, and giving the unneeded items to your dear neighbours, if you can save money in this way.

Note that all prices  $P$  given in the input 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

### 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 \$40.00