Competitive Programming SS24

Submit until end of contest

Problem: coinchange (3.0 second timelimit)

Bob is really afraid of germs and, thus, hates touching anything that hasn't been thoroughly disinfected. He especially hates coins. Just think of where those might have been and what forms of life might be evolving in your wallet right now!

Unfortunately, Bob still owes Jimmy X€ and Jimmy wants his money back. As electronic payment is not an option for Jimmy, Bob wants to figure out how to pay back Jimmy while exchanging a minimum number of coins.

Assume that Bob's debt was 4€. Then Bob would have to give Jimmy two 2€ coins and therefore two coins would have to be exchanged. Now assume that the debt was 4.99€. In this case it would be best if Bob gave Timmy a 5.00€ bank note[1] and Timmy returned a 0.01€ coin. Again, two coins would be exchanged. If the debt was 19.01€, then Bob should give Jimmy a 20€ bank note as well as a 0.01€ coin and return a 1€ coin.

Help Bob by writing a program that computes the minimum number of coins that have to be transferred given the values of the coins and Bob's debt. You may assume that Bob and Jimmy have an infinite number of every coin at their disposal.

Input The first line contains the number of test cases t ($1 \leq t \leq 10$). The first line of every test case contains two integers X ($0 \leq X \leq 10^7$) and C ($1 \leq C \leq 10$) in this order. The debt is given by X and C is the number of different coin values.

Next a line with C integers separated by spaces follows. These are the values of the coins. They are given in ascending order and you may assume that no two coins exist with the same value. Also, there will always be a coin with value 1 and the maximum coin value will not exceed 3000.

The sum over X for all testcases does not exceed 10^7 ($\sum_{i=1}^{t} X_i \leq 10^7$)

Output For every test case output a single line containing the minimum number of coin exchanges needed.

[1] Bob hates bank notes just as much as he hates coins, so there is no point in discerning between the two.

Sample input

```
3
400 5
1 100 200 500 2000
499 5
1 100 200 500 2000
1901 5
1 100 200 500 2000
```

Sample output

```
2
2
3
```