Problem A: Firewood

Advanced Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

Little Jony lives in a small village. Winter is coming, so his grandmother asked him to go to the nearby forest and chop some firewood. In said forest there is a magical clearing where there always lies a bundle of logs ready to take. Obviously, Jony wants to go just there.

The only problem with that is that it's pretty far off from the village and Jony's walking speed in the woods is much lower than on the fields that surround the village and reach right up to the edge of the woods.

- We model the region as the Cartesian plane
- The village is located at $(0, 1)$.
- The clearing is at $(1, 0)$.
- The boundary between the forest and field is the horizontal line $y = a$, where $a \in [0, 1]$ is part of the input.
- Jony's walking speed is v_p on the fields and v_f in the forest. On the boundary he can walk in the forest as well as in the field, however he wishes.

Find the point on the boundary line where Jony should enter the forest (i.e. leave the boundary behind, not walk along it) to reach the magic clearing as quick as possible.

Input

First line contains two positive integers – v_p and v_f ($1 \leq v_p, v_f \leq 10^5$). The second line contains the number $a \in [0, 1]$ described above.

Output

Output a single number – the coordinate on the x-axis of the point where Jony should enter the forest. Your precision should be at least 10^{-6}.

Sample input and output

Input	Output
5 3 0.4	0.783310604
5 5 0.5	0.500000000

Problem B: Bulls

Advanced Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

A local cow breeder begged you to help him: Even considering it is mating season, his bulls are being unusually aggressive towards one another, even when they are physically separated but kept in close proximity. Since he can't figure out why that is, he decided to separate them as far as possible until their minds have settled.

For that he wants to use his old, currently unused stalls, which are all located in the same street, here modeled as an axis of coordinates. He already made sure that the stalls are still intact but now he needs your help to figure out how far he can separate the bulls, i.e. what the minimum distance will be if he distributes them onto the stalls optimally.

Input

The input consists of

- one line containing N ($3 \leq N \leq 10^4$) – the number of stalls and K ($2 \leq K < N$) – the amount of bulls
- one line containing N non-negative integers in ascending order – the coordinates of the stalls (they are not exceeding 10^9).

Output

Output a single number – the largest minimum distance (w.r.t. the stalls' coordinates) between neighboring bulls that can be achieved by distributing all of them onto the stalls.

Sample input and output

Input	Output
5 3 1 2 3 100 1000	99

Problem C: Close Trains

Advanced Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

In consequence of a series of tragic events on the rails between the two cities of the small country of Dreamland, its government finally acknowledged that the train schedule needed to change. A detailed look on the rails' condition and environment showed that the optimal solution is the following schedule: Each train starts with pace v_1 meters per minute and holds it for T_1 minutes, the next T_2 minutes it travels at v_2 meters per minute and so on, until, finally, the last T_N minutes are traveled at v_N meters per minute. In some of those time intervals the train could stand still (pace 0).

In addition to the new schedule, the government also came up with a new safety guideline: The distance between two trains traveling one after another should be at least L meters at all times. Find the minimum amount of time a train has to wait after his predecessor's start in order to adhere to the safety guideline, assuming both trains meticulously follow the schedule described above.

Input

The input consists of

- one line containing two integers L ($100 \leq L \leq 10^4$) – the smallest permitted distance – and N ($1 \leq N \leq 10^3$) – the number of segments in the schedule
- N lines each containing a pair of integers T_i and v_i ($1 \leq T_i \leq 10^3$, $0 \leq v_i \leq 10^3$). The pairs are of course given in the order of indexation.

Output

Output the smallest permitted time difference between the starts of two successive trains with precision at least 10^{-3}.

Sample input and output

Input	Output
1000 4 10 0 30 80 15 0 20 100	27.500000

Problem D: Largest square

Advanced Algorithms for Programming Contests

Restrictions

Time: 5 seconds
Memory: 512 MB

Problem description

There is an $N \times N$ mosaic of square solar cells. Each solar cell is either good or bad. There are W bad cells. You need to find the largest square within the mosaic containing at most L bad cells.

Input

The input will begin with a number T ($1 \leq T \leq 20$), the number of test cases, on a line by itself. Then follow the T test cases, each consisting of

- one line containing N, W and L ($1 \leq N \leq 2000, 1 \leq W \leq 5 \cdot 10^4$, $0 \leq L \leq W$) – the side length of the grid, the number of bad cells and the maximum amount of bad cells allowed in the subgrid, respectively
- W lines, each containing integers a and b ($1 \leq a, b \leq N$), representing the coordinates of a location of one of the bad solar cells.

Output

For each input instance, the output should be a single integer representing the area of the largest square that contains no more than L bad solar cells.

Sample input and output

Input	**Output**
1 4 3 1 1 1 2 2 2 3	4

Clarification

In the given sample, the mosaic is 4×4, and contains the following arrangement of good and bad cells ('`G`' represents good and '`B`' bad):

```
BGGG
GBBG
GGGG
GGGG
```

Several 2×2 squares at the bottom contain no bad solar cells, but all 3×3 squares contain at least two bad solar cells.

Problem E: Eulerian Inference

Advanced Algorithms for Programming Contests

Restrictions

Time: 2 seconds
Memory: 512 MB

Problem description

You are to guess n, a positive integer not exceeding 10^4, based on at most 42 queries asking for $\phi(\lfloor a/b \cdot n \rfloor)$ for some positive integers $a, b \leq 10^4$, where ϕ is Euler's totient function and $\lfloor \cdot \rfloor$ is the floor function. There are several testcases.

Hint: It might make sense to look up properties of Euler's totient function, e.g. in the lecture slides of Algo 1 topic 12 (number theory), which were uploaded to the "Supplementary Material" section of this course's Stud.IP.

Interaction Protocol

First you are given T ($1 \leq T \leq 100$) – the number of testcases you need to answer. For each testcase, the interaction works as follows:

To ask for $\phi(\lfloor a/b \cdot n \rfloor)$, simply output "? a b" and read the answer as an integer. To submit your guess m, output "! m". After submitting a guess you should move on to the next testcase or terminate if it was the last one. For technical reasons, most misbehavior yields the verdict `Run Error`.

Please note that, since $\phi(0)$ is undefined, if you pose a query "? a b" where $\lfloor a/b \cdot n \rfloor = 0$ (and a, b valid), there are no guarantees about what the answer will be (but it won't cause a `Run Error`).

Sample input and output

Input	Output
1 2 1	 ? 1 2 ? 1 3 ! 6

Problem F: Fun Array

Advanced Algorithms for Programming Contests

Restrictions

Time: 4 seconds
Memory: 512 MB

Problem description

Alice has conceived of a very fun array consisting of n integers $a_1, ..., a_n$. Of course Bob wants to know it, but she refuses to share it with him. Instead, she will just answer at most n questions of a certain kind: In each question, Bob needs to name k pairwise distinct indices $i_1, ..., i_k$ in the range from 1 to n and in response Alice will tell him the sum of the corresponding elements, $a_{i_1} + ... + a_{i_k}$. Can you help Bob to figure out the array?

Interaction Protocol

First you are given the parameters n and k ($2 \leq n \leq 1000, 1 \leq k < n$). Then you may ask up n questions. Each of those questions must be posed in the form "? i_1 i_2 ... i_k" ($1 \leq i_j \leq n, i_{j_1} \neq i_{j_2} \forall j_1 \neq j_2$), as in the sample below. In response to each question, the answer will be given to you as a single integer on a new line. Finally, you should communicate the fully deduced array, using the format "! a_1 a_2 ... a_n". After doing this, your program should terminate immediately. Posing an invalid question will cause the verdict `Run Error`. Printing any wrong number in the final line of output will cause the verdict `Wrong Answer`. It is guaranteed that all entries of Alice's array lie in the range from -10^6 to 10^6.

Sample input and output

Input	Output
4 2 17 19 18	 ? 1 2 ? 3 4 ? 4 2 ! 7 8 9 10

Note that the questions and their answers in this sample did not actually determine the array, e.g. $a = [17, 0, 1, 18]$ would also have been possible.