ICPC International Collegiate Programming Contest // 2025-2026

The 2025-2026 ICPC
Latin America Contests

ICPC Latin American Regional Contests – 2025

November 07, 2025

Warmup Session

This problem set contains 4 problems; pages are numbered from 1 to 8.

This problem set is used in simultaneous contests with the following participating countries:

Antigua y Barbuda, Argentina, Bolivia, Brasil, Chile, Colombia, Costa Rica, Cuba, Ecuador, El Salvador, Guatemala, México, Perú, Puerto Rico, República Dominicana, Trinidad y Tobago, Uruguay and Venezuela

v1.0

General information

Unless otherwise stated, the following conditions hold for all problems.

Program name

1. Your solution must be called *codename*`.c`, *codename*`.cpp`, *codename*`.java`, *codename*`.kt`, *codename*`.py3`, where *codename* is the capital letter which identifies the problem.

Input

1. The input must be read from standard input.
2. The input consists of a single test case, which is described using a number of lines that depends on the problem. No extra data appear in the input.
3. When a line of data contains several values, they are separated by *single* spaces. No other spaces appear in the input. There are no empty lines.
4. The English alphabet is used. There are no letters with tildes, accents, diaereses or other diacritical marks (ñ, Ã, é, Ì, ô, Ü, ç, etcetera).
5. Every line, including the last one, has the usual end-of-line mark.

Output

1. The output must be written to standard output.
2. The result of the test case must appear in the output using a number of lines that depends on the problem. No extra data should appear in the output.
3. When a line of results contains several values, they must be separated by *single* spaces. No other spaces should appear in the output. There should be no empty lines.
4. The English alphabet must be used. There should be no letters with tildes, accents, diaereses or other diacritical marks (ñ, Ã, é, Ì, ô, Ü, ç, etcetera).
5. Every line, including the last one, must have the usual end-of-line mark.

Problem A – Append and Panic!

Today is Gabriel's first day at his new job, and he has been given his first task. He needs to read a string made up of uppercase letters from a file, sort the letters in the string alphabetically, filter out repeated letters, and then write the result back to the original file. For instance, sorting the string "`ICPC`" would produce "`CCIP`", which would become "`CIP`" after removing repeated letters. Easy, right?

However, Gabriel made a tiny mistake. Instead of overwriting the file with the filtered string, he accidentally appended it to the file. Now, the file is corrupted, containing the original string followed by the sorted, duplicate-free version of it, and Gabriel is in a bit of a panic.

Given the content of the corrupted file, can you determine the length of the original string? Gabriel is confident that with this information, he will be able to complete his assigned task.

Input

The input consists of a single line that contains a string S made up of uppercase letters ($2 \leq |S| \leq 2000$), which is the concatenation of the original (uncorrupted) string t and the sorted, duplicate-free version of t.

Output

Output a single line with an integer indicating the length of t.

Sample input 1	Sample output 1
`ICPCCIP`	`4`

Sample input 2	Sample output 2
`ABEDCCCABCDE`	`7`

Sample input 3	Sample output 3
`ZZ`	`1`

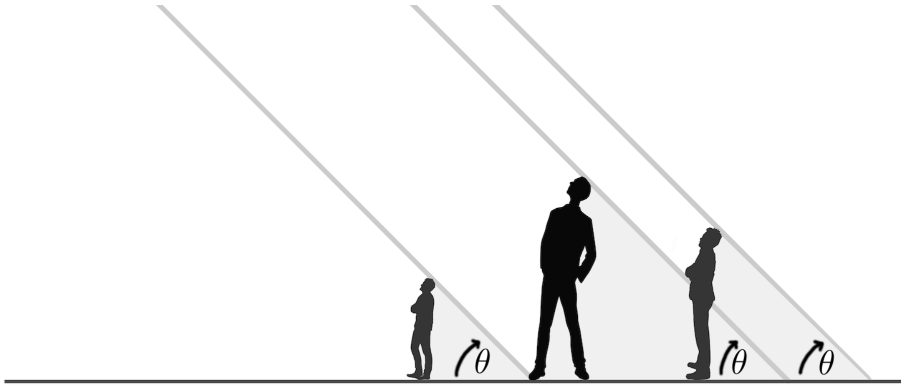
This page would be intentionally left blank if we would not wish to inform about that.

Problem B – Jigsaw of Shadows

They finally did it! The flat-earthers managed to teleport themselves to an idealized flat world where no one can make fun of them anymore.

In this world, there is a perfectly straight, infinite road that runs along the x-axis. Instead of a sun, they have something more illuminating: a gigantic flashlight elevated above the road, positioned infinitely far away to the west (negative x-axis). This flashlight beams light at a precise angle with respect to the ground, illuminating the entire road.

There are N flatlanders standing proudly at distinct positions along the road. As the light strikes each of them, it casts a shadow extending eastward (toward the positive x-axis).



Eager to show off their flat-world knowledge, the citizens want to calculate how much of the road is covered by their shadows. However, with shadows potentially overlapping, they need your help to figure it out. Given the positions of the flatlanders and their heights, can you calculate the total length of the road covered by their shadows?

Input

The first line contains two integers θ ($10 \le \theta \le 80$) and N ($1 \le N \le 10^5$), indicating respectively the angle of the light beams and the number of flatlanders on the road. The angle is measured in degrees, clockwise from the ground to the light beams.

Each of the next N lines contains two integers X ($0 \le X \le 3 \cdot 10^5$) and H ($1 \le H \le 1000$), indicating that a flatlander of height H is located at position X along the road. It is guaranteed that no two flatlanders share the same location.

Output

Output a single line with the total length of the road covered by the shadows of all flatlanders. The output must have an absolute or relative error of at most 10^{-4}.

Sample input 1	Sample output 1
45 3 50 150 0 100 100 200	300

Sample input 2	Sample output 2
60 3 50 150 0 100 100 200	215.47

Sample input 3	Sample output 3
30 3 50 150 0 100 100 200	446.41016

Problem C – Finding Privacy

You surely agree that the best place to use a restroom is at home. However, sometimes people have no choice but to use a public restroom, where toilets are often arranged side by side in a single row. Aiming for some privacy, each person who enters such restroom will choose an unoccupied toilet that has no occupied toilets on its sides.

Suppose that K people arrive at a public restroom with N initially unoccupied toilets arranged in a row. Determine if it's possible that each of the K people chooses a toilet with no occupied toilets on its sides, and an additional person would not be able to find an unoccupied toilet meeting this privacy condition. People choose toilets one by one, and each chosen toilet is immediately occupied before the next person is allowed to choose.

Input

The input consists of a single line that contains two integers K and N ($1 \leq K \leq N \leq 10^6$), indicating respectively the number of people and the number of toilets.

Output

Output a single line with a string of length N if the K people can choose toilets in a way that prevents an additional person from finding an available toilet with the required privacy. In this case the i-th character of the string must be the uppercase letter "`X`" if the i-th toilet is chosen, and the character "`-`" (hyphen) otherwise. Toilets are chosen as it is described in the statement. If there are multiple solutions, output any of them.

If toilets cannot be chosen as requested, output a line with the character "`*`" (asterisk) instead.

Sample input 1	Sample output 1
`1 5`	`*`

Sample input 2	Sample output 2
`2 5`	`-X-X-`

Sample input 3	Sample output 3
`3 5`	`X-X-X`

Sample input 4	Sample output 4
`4 5`	`*`

Sample input 5	Sample output 5
`5 5`	`*`

Sample input 6	Sample output 6
`2 5`	`-X--X`

This page would be intentionally left blank if we would not wish to inform about that.

Problem D – Hidden Treasure

This problem is interactive.

There is a hidden treasure at integer coordinates (r, c) on the square bounded by $(0, 0)$ and (N, N), that is, $0 \leq r, c \leq N$. Your task is to find the coordinates of the treasure by asking questions to the interactor.

In each question, you may pick integer coordinates (x, y) $(0 \leq x, y \leq N)$, and the interactor will respond with the Manhattan distance from (x, y) to the treasure, i.e. $|r - x| + |c - y|$.

You may ask at most 5 questions. When you have determined the coordinates of the treasure, output them in the format described below and terminate your program.

Interaction

First, read a single integer N $(1 \leq N \leq 10^5)$, the upper bound for the coordinates of the treasure.

You may then ask at most 5 questions to the interactor. Each question must be asked in the format "`? x y`" (without quotes), where $0 \leq x, y \leq N$.

After each query, read a single integer d, the Manhattan distance from the queried point to the treasure.

When you have determined the coordinates of the treasure (r, c), print "`! r c`" (without quotes) and terminate your program.

Exceeding the number of allowed questions or printing an invalid query will result in a `Wrong Answer` verdict.

The interactor is **not adaptive**, meaning the treasure's position is fixed before the interaction starts and does not change based on your queries.

After each write to the output, remember to flush the output buffer. Otherwise, you may receive the verdict `Time Limit Exceeded`. To flush the buffer, use:

- `fflush(stdout)` in `C`.
- `cout.flush()` in `C++`.
- `sys.stdout.flush()` in `Python`.
- `System.out.flush()` in `Java`.
- `System.out.flush()` in `Kotlin`.

Sample interaction 1

Read	Write
3	
	? 0 0
5	
	? 1 0
4	
	? 2 0
3	
	? 3 0
4	
	? 0 3
2	
	! 2 3

Explanation of sample 1:

First, the program reads $N = 3$, indicating that the treasure lies within coordinates $(0, 0)$ and $(3, 3)$.
The first four queries fix $y = 0$ and test $x = 0, 1, 2, 3$. The interactor's responses are $5, 4, 3, 4$, indicating that the treasure is at coordinates $(2, y)$ for some y, since the distances decrease until $x = 2$ and then increase again.
The fifth query checks $(0, 3)$ and receives a response of 2. From this we deduce:

$$|2 - 0| + |c - 3| = 2 \Rightarrow |c - 3| = 0 \Rightarrow c = 3.$$

Hence, the treasure is at $(2, 3)$, and the program outputs:

`! 2 3`.