DIU Unlock the Algorithm Programming Contest Spring-25 [Final]

https://toph.co/c/diu-unlock-the-algorithm-spring-25-final

Toph

Schedule

The contest will run for **3h30m0s**.

Authors

The authors of this contest are faf_7s, fahimcp495, Hamza_28, kazi_amir, md_husain, Mohimenul, nahid_1, Piyash_Basak, Saimur, sakibsidha, and sourov.cse.

Rules

This contest is formatted as per the official rules of ICPC Regional Programming Contests.

You can use Bash 5.2, Brainf*ck, C# Mono 6.0, C++17 GCC 13.2, C++20 Clang 16.0, C++20 GCC 13.2, C++23 GCC 13.2, C11 GCC 13.2, C17 GCC 13.2, C23 GCC 13.2, Common Lisp SBCL 2.0, D8 11.8, Erlang 22.3, Free Pascal 3.0, Go 1.22, Grep 3.7, Haskell 8.6, Java 1.8, Kotlin 1.9, Kotlin 2.0, Lua 5.4, Node.js 10.16, Perl 5.30, PHP 8.3, PyPy 7.3 (3.10), Python 3.12, Ruby 3.2, Rust 1.84, Swift 5.3, and Whitespace in this contest.

Be fair, be honest. Plagiarism will result in disqualification. Judges' decisions will be final.

Notes

There are 9 challenges in this contest.

Please make sure this booklet contains all of the pages.

If you find any discrepencies between the printed copy and the problem statements in Toph Arena, please rely on the later.

A. Robot

You are given an $N * M$ grid where each cell is either **land** (`.`) or **sea** (`#`). There are K robots labeled from `1` to K. The $i-th$ robot starts at cell (x_i, y_i) (guaranteed to be land). Multiple robots can occupy the same cell.

Each robot can move **up, down, left, or right** to adjacent land cells but cannot cross sea cells. The goal is for a robot to reach the **destination cell**(N, M)(also land) to deliver a message.

You are given Q queries. For each query:

- An integer $i(1 \leq i \leq K)$ which means that the $i-th$ robot is **unable to function**.
- You must determine the minimum time taken among all **remaining robots** who can reach (N, M).
- If no robot can reach the destination, output `-1`.

Input

- First line: N and M $(1 \leq N, M \leq 1000)$.
- Next N lines: The grid (each row as a string of `.` or `#`).
- Next line: K $(1 \leq K \leq 10^5)$.
- Next K lines: x_i $(1 \leq x_i \leq N)$and y_i $(1 \leq y_i \leq M)$.
- Next line: Q $(1 \leq Q \leq 10^5)$.
- Next Q lines: An integer i $(1 \leq i \leq k)$.

Output

- For each query, output one integer:
- The time taken or `-1` if impossible.

Samples

Input	Output
3 3 ... .#. ... 3 1 1 1 3 3 1 2 2 3	2 2

Explanation:

- **Grid:** The sea at $(2, 2)$ blocks diagonal paths.
- **Robots:**
  - Robot 1: $(1, 1)$→ Path: $(1, 1) \to (1, 2) \to (1, 3) \to (2, 3) \to (3, 3)(Time = 4)$.
  - Robot 2: $(1, 3)$ → Path: $(1, 3) \to (2, 3) \to (3, 3)(Time = 2)$.
  - Robot 3: $(3, 1)$ → Path: $(3, 1) \to (3, 2) \to (3, 3)(Time = 2)$.
- **Queries:**
  1. Robot 2 is broken. The fastest remaining robot is Robot 3 $(Time = 2)$.
  2. Robot 3 is broken. The fastest remaining robot is Robot 2 $(Time = 2)$.

B. AC Installation Plan

Daffodil International University (DIU) wants to install **one air conditioner (AC) in each classroom** to keep students cool and focused. Each AC consumes a fixed number of **electricity units per month**.

The university has a **fixed monthly budget**, and electricity is billed using a **tiered pricing system** — the more units you use overall, the more expensive each unit becomes.

Your job is to help DIU figure out **how many ACs they can install** without crossing their electricity budget.

Electricity Cost Structure:

- For the first **50 units** → 10 taka per unit
- For units **51 to 80** → 15 taka per unit
- For units **81 to 100** → 20 taka per unit
- For any unit **above 100** → 25 taka per unit

Example: If **3 ACs** are installed, and each AC uses **40 units**, then total usage = `3 × 40 = 120 units`
The cost is calculated like this:

- First 50 units → 50 × 10 = 500
- Next 30 units → 30 × 15 = 450
- Next 20 units → 20 × 20 = 400
- Remaining 20 units → 20 × 25 = 500
- **Total cost = 500 + 450 + 400 + 500 = 1850 taka**

Input

Two integers X and Y

X = DIU's total electricity budget in taka
Y = electricity units one AC uses per month

$1 \leq X, Y \leq 10^{12}$

Output

Print **one integer** — the **maximum number of ACs** DIU can install **without exceeding the budget**.

Samples

Input	Output
3000 40	4

C. Know your Siblings

You are given a rooted tree with N nodes rooted at node 1. Each of the $N-1$ edges connects two nodes. For every node from 2 to N, you need to determine how many siblings it has. (A sibling is a node that shares the same parent)

Input

- $N (2 \leq N \leq 10^5)$: the number of nodes in the tree.
- $N-1$ lines follow, each containing two integers u and v representing an undirected edge between nodes u and v.

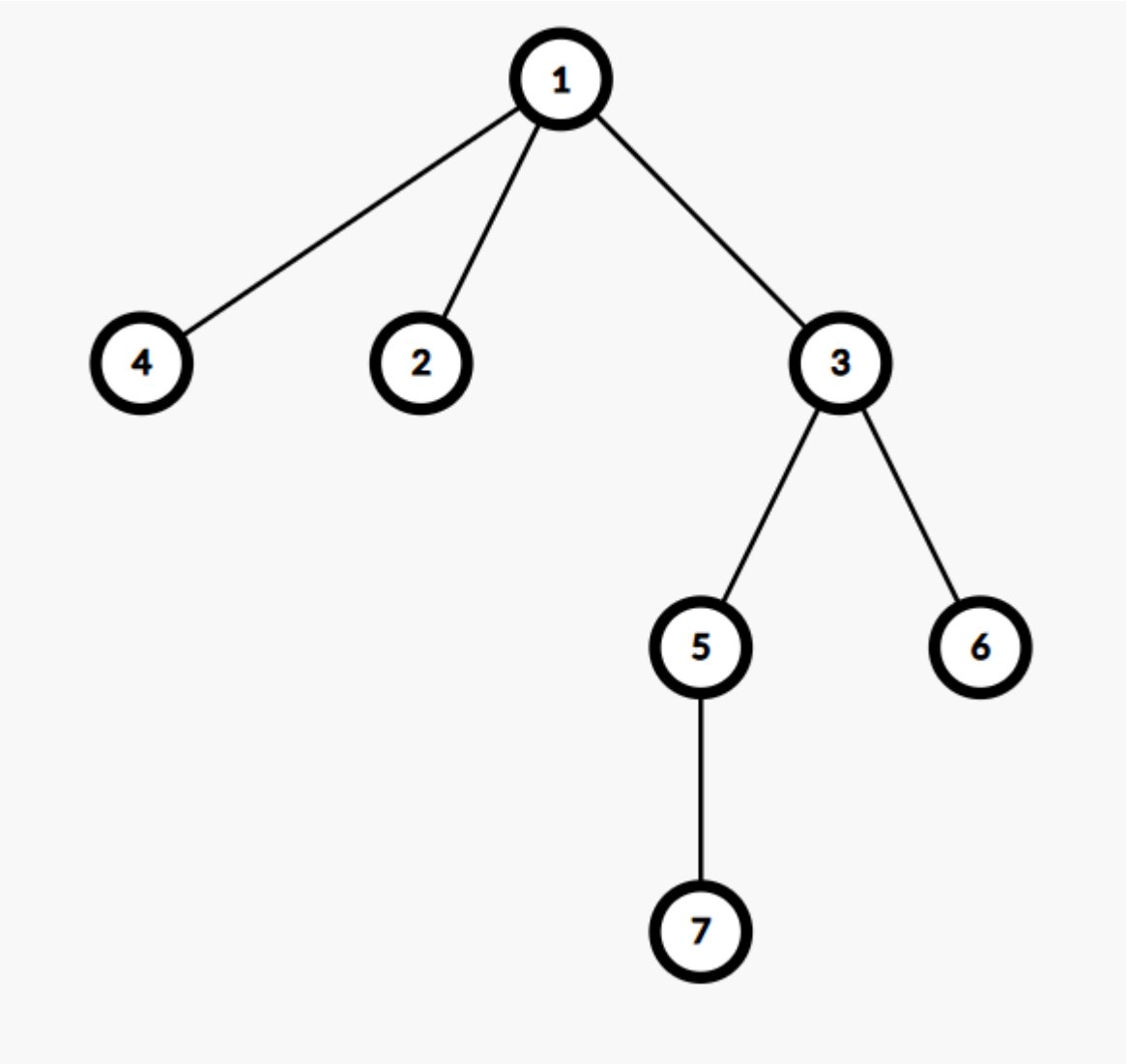
Output

Print $N-1$ integers on a single line: the number of siblings for node 2, node 3, ..., node N.

Samples

Input	Output
7 1 2 1 3 1 4 3 5 3 6 5 7	2 2 2 1 1 0

Input	Output



Explanation:

- Node 2, 3, and 4 all have the same parent (node 1) → each has 2 siblings.
- Node 5 and 6 have same parent (node 3) → each has 1 sibling.
- Node 7 is the only child of node 5 → 0 siblings.

D. Narcissistic Number

A narcissistic number is a special kind of non-negative integer. It is defined as a number N that is the sum of its own digits, each raised to the power equal to the number of digits in the number.

Formally, for a number N with D digits:

$$N = \sum_{i=1}^{D} (Digit_i)^D$$

where each $Digit_i$ is a digit of N, and D is the total number of digits.

In simpler terms:

- Break the number into its digits.
- Raise each digit to the power of the number of digits.
- Add up these powered digits.
- If the result equals the original number, it is a narcissistic number.

Examples:

153 is a narcissistic number because:

$1^3+5^3+3^3=1+125+27=153$

9474 is a narcissistic number because:

$9^4+4^4+7^4+4^4$ =$6561+ 256+2401 +256= 9474$

14 is not a narcissistic number because:

$1^2+4^2=1+16=17$

Your task is to find the smallest non-negative narcissistic number.

Input

There is no input for this problem.

Output

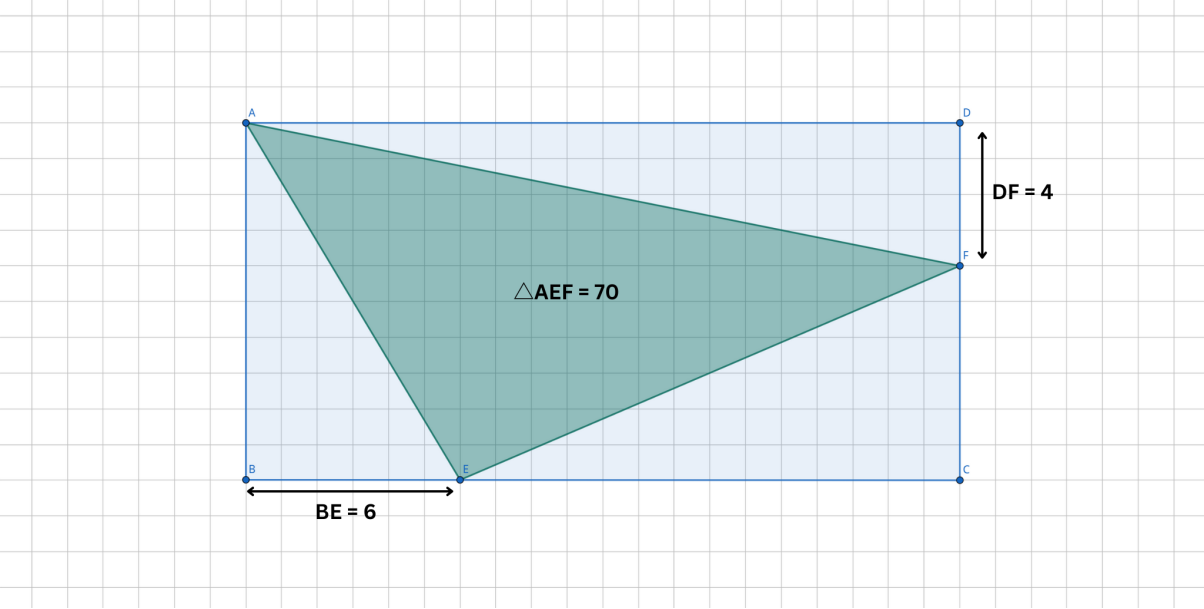
You have to print a single integer number X. Here, X represents the smallest non-negative narcissistic number.

E. Area Recovery

In rectangle $ABCD$, points E and F lie on sides BC and CD respectively. You are given:

- The length of segment BE,
- The length of segment DF,
- The area of triangle $\triangle AEF$.

Look at the image for Clarification.



Using this information, determine the area of rectangle $ABCD$.

Input

The input contains Three integers separated by spaces A,B and X.

A — the length of segment BE ($1 \leq A \leq 10^9$)

B — the length of segment DF ($1 \leq B \leq 10^9$)

X — Area of triangle $\triangle AEF$ ($1 \leq X \leq 10^{18}$)

It is guaranteed that $BE < BC$and $DF < DC$.

Output

Your task to print a integer which is The area of rectangle $ABCD$.

It can be proved that the area of a rectangle always be an integer number under given constrains.

Samples

Input	Output
6 4 70	164

Be careful about the newline('\n') at the end.

F. Minimize Difference

Given an array A of n integers, choose three elements (with different indices) such that the maximum difference between any two of them is minimized.

Input

The first line contains one integers $n(3 \leq n \leq 2 \cdot 10^5)$— the number of elements in the array.

The second line contains n integers $A_1\ A_2 \ldots\ A_n(1 \leq A_i \leq 10^9)$

Output

Print a single integer — the minimum possible value of that maximum difference.

Samples

Input	Output
3 3 1 2	2

G. MindSync Neural System

In the year 3025, the ACM community of Daffodil International University launched the revolutionary **MindSync Neural System** — a digit-powered grid where each cell represents a brain node's signal strength, ranging from 0 (fully idle) to 9 (hyperactive).

This matrix-like neural structure was designed with two strict synchronization rules for optimal brainwave alignment:

1. All **horizontally adjacent** nodes must have the same signal strength.
2. All **vertically adjacent** nodes must have different signal strengths.

Unfortunately, after a cosmic ray burst corrupted the grid, the signal strengths are now in disarray. The matrix needs urgent reprogramming.

You are the neural architect tasked with re-aligning the MindSync matrix by changing the **minimum** number of signal strengths (digits) to make it good again.

Input

The first line contains two integers m and n, representing the number of rows and columns in the matrix, respectively.

Each of the next m lines contains a string of n digits (from $0\ to\ 9$), representing the current state of the matrix.

Constraints

- $1 \leq m, n \leq 1000$
- The input matrix contains only characters $0\ to\ 9$.

Output

Print a single integer — the **minimum** number of digits that need to be changed to make the matrix satisfy the **MindSync** conditions.

Samples

Input	Output
3 3 1 2 2	2

Input	Output
3 3 3 1 1 3	

H. Frequency Showdown

You're given an array A of N positive integers. Your task is to identify the K most frequent elements in this array and print them in descending order of their frequency. If two or more elements have the same frequency, print the smaller element first.

But wait, there's a twist! The number of distinct elements may be less than K. In such cases, print all the available elements in the correct order.

Input

The first line contains two space-separated integers N and K $(1 \leq K \leq N \leq 10^5)$ — the number of elements in the array and the number of top frequent elements to print respectively.

The second line contains N space separated integers $A_1, A_2, ..., A_n (1 \leq A_i \leq 10^9)$ representing the elements of the array.

Output

Print the K most frequent elements from the array in the correct order. If there are fewer than K distinct elements, print all of them.

Samples

Input	Output
4 4 1 1 2 3	1 2 3

Input	Output
12 3 1 2 3 4 1 2 1 4 6 6 8 6	1 6 2

Be careful about the newline ('\n') at the end.

I. Yaaay, Party!

Monika is hosting a day-long birthday party and has invited N friends. Each friend can arrive and depart at different times during the day.

- When a friend arrives, they immediately take a chair and sit down at the moment of arrival.
- No friend(including Monika) can be left standing — a chair must always be available for every friend present at the party at any moment.
- When a friend leaves at any moment x, their chair will be available for someone else to sit in at moment $x + 1$

  For example, if someone leaves at $10ms$, the chair will be available to sit in from $11ms$
- Initially, Monika had 2 chairs at her house.

Monika knows all of her friends' arrival and departure times before the party starts. You are given two arrays:

- $a_1\ a_2\ a_3 \ldots\ a_n$ — where a_i is the time (in milliseconds) when the $i-th$ friend arrives.
- $b_1\ b_2\ b_3 \ldots\ b_n$ — where b_i is the time (in milliseconds) when the $i-th$ friend leaves.

Your task is to determine the minimum number of additional chairs Monika must buy so that no friend has to stand at any time.

Input

The input is provided in the following format through standard input.

- The first line of input contains one integer N representing the number of friends attending the party
- The second line contains N space-separated integers denoting the arrival time of each friend
- The third line contains N space-separated integers denoting the leaving time of each friend

Constraints:

$0 \leq N \leq 10^5$

$1 \leq a_i < b_i \leq 10^6$

Output

Print the minimum number of chairs Monika needs to buy.

Samples

Input	Output
5 2 2 3 5 18 3 4 4 6 20	2