icpc International Collegiate Programming Contest

# The 2024 ICPC Asia Yokohama Regional Contest

# Problem Set

icpc.foundation

icpc Asia Yokohama

JETBRAINS

icpc global sponsor
programming tools

HUAWEI

icpc diamond
multi-regional sponsor

# Problem A
## Ribbon on the Christmas Present
Time Limit: 2 seconds

You are preparing a ribbon to decorate the Christmas present box. You plan to dye the ribbon, initially white, to make a stripe pattern of different shades of red. The ribbon consists of a number of sections, each of which should be dyed as planned.

You want to prepare the ribbon with the least number of dyeing steps. Contiguous sections of the ribbon can be dyed in one step with the same shade of red. A ribbon section already dyed with some shade of red can be overdyed with dyestuff of a darker shade; it is colored with that darker shade. Overdyeing with a lighter shade is, however, not allowed. As the ribbon is initially white, all the sections must be dyed at least once.

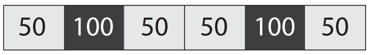

Figure A.1. Stripe Pattern of Sample Input 1

Figure A.1 shows the pattern of Sample Input 1. The ribbon has six sections and the numbers in the sections mean the levels of shades to be dyed. Larger numbers mean darker shades. This can be made by three dyeing steps:

1. dye the entire ribbon with red dyestuff of shade level 50,
2. dye the second section from the left with darker shade dyestuff of level 100, and then
3. dye the fifth section with dyestuff of level 100.

Write a program that computes the least number of dyeing steps to make the planned stripe pattern.

### Input

The input consists of a single test case of the following format.

$$n$$
$$d_1\ d_2 \cdots\ d_n$$

The test case starts with an integer $n$ $(1 \leq n \leq 100)$, the number of sections of the ribbon. The second line contains $n$ integers, $d_1, d_2, \ldots, d_n$, describing the planned shade levels of the $n$ sections. Here, $d_i$ means the planned shade level of the $i$-th section, which is between 1 and 100, inclusive, larger meaning darker.

### Output

Output a line containing the least number of dyeing steps to make the planned stripe pattern.

| **Sample Input 1** | **Sample Output 1** |
|---|---|
| 6<br>50 100 50 50 100 50 | 3 |

### Sample Input 2

```
5
1 2 3 2 1
```

### Sample Output 2

```
3
```

### Sample Input 3

```
5
3 2 1 2 3
```

### Sample Output 3

```
5
```

### Sample Input 4

```
10
1 20 100 1 20 20 100 100 20 20
```

### Sample Output 4

```
5
```

### Sample Input 5

```
5
10 60 100 30 10
```

### Sample Output 5

```
4
```

# Problem B
## The Sparsest Number in Between
Time Limit: 2 seconds

You are given a pair of positive integers $a$ and $b$ ($a \leq b$). Among those integers between $a$ and $b$, inclusive, your task is to find the *sparsest* one, that is, the one with the least number of `1`'s in its binary representation. If there are two or more such integers, you should find the smallest among them.

Suppose, for instance, that $a = 10$ and $b = 13$. The integers between $a$ and $b$, inclusive, are 10, 11, 12, and 13, and their binary representations are `1010`, `1011`, `1100`, and `1101`, respectively. Thus, in this case, the answer is 10, since 10 and 12 have the least number of `1`'s in their binary representations and 10 is smaller than 12.

### Input

The input consists of a single test case of the following format.

$$a\ b$$

Here, $a$ and $b$ ($a \leq b$) are integers between 1 and $10^{18}$, inclusive.

### Output

Output a line containing the smallest among the sparsest integers between $a$ and $b$, inclusive.

| Sample Input 1 | Sample Output 1 |
|---|---|
| `10 13` | `10` |

| Sample Input 2 | Sample Output 2 |
|---|---|
| `11 15` | `12` |

| Sample Input 3 | Sample Output 3 |
|---|---|
| `11 20` | `16` |

| Sample Input 4 | Sample Output 4 |
|---|---|
| `1 1000000000000000000` | `1` |

| Sample Input 5 | Sample Output 5 |
|---|---|
| `9876543210 9876543210` | `9876543210` |

# Problem C
## Omnes Viae Yokohamam Ducunt?
### Time Limit: 3 seconds

*“Omnes viae Romam ducunt”* is an old Latin proverb meaning “all roads lead to Rome.” It is still desirable to have access to the capital from all the regions of a country.

The Kingdom of Kanagawa has a number of cities, including the capital city, Yokohama. The Ministry of Transport of the Kingdom is now planning to construct a highway network, connecting all those cities.

There are a number of candidate highway segments, each of which directly connects two cities. A highway network is a set of highway segments chosen from the candidates. The following are required for the highway network.

- All the cities should be connected via highway segments in the network, directly or indirectly.
- To save the budget, the minimum number of segments should be chosen. In other words, the highway network should not be redundant; the path connecting any pair of cities should be unique.

The highway network should be made resistant to natural disasters, with the limited budget. The emphasis is placed on accessibility to and from the capital city, Yokohama. As the network is planned to be non-redundant, when one segment becomes unavailable due to a natural disaster, some of the cities become inaccessible from Yokohama.

We want to minimize the *total risk severity,* defined as follows.

The cities in the Kingdom have different populations and economic scales, based on which, the cities are assigned certain *significance values*. Given a highway network, the *damage* suffered from a natural disaster on a single segment in the network is estimated by the sum of the significance values of such cities made inaccessible from Yokohama.

*Vulnerabilities* to natural disasters are assessed for all the candidate segments. The *risk severity* of a segment is calculated as the product of its estimated damage and vulnerability. The total risk severity of the network is estimated as the sum of the risk severities of all the segments in the network.

Your task is to determine the minimum total risk severity by appropriately designing the highway network.

## Input

The input consists of a single test case of the following format.

$$
\begin{array}{l}
n\ m \\
p_1 \cdots p_n \\
u_1\ v_1\ q_1 \\
\vdots \\
u_m\ v_m\ q_m
\end{array}
$$

The first two integers $n$ and $m$ ($2 \le n \le 10^5$, $1 \le m \le 3 \times 10^5$) describe the numbers of cities and highway segment candidates, respectively. The cities are numbered from 1 to $n$, with Yokohama numbered 1. The second line contains $n$ integers $p_1, \ldots, p_n$, where each $p_i$ ($1 \leq p_i \leq 1000$) represents the significance value assigned to the city numbered $i$.

The following $m$ lines describe the candidate highway segments. The $j$-th line of them contains three integers $u_j$, $v_j$, and $q_j$ ($1 \leq u_j < v_j \leq n, 1 \leq q_j \leq 10^6$), meaning that the segment candidate connecting cities numbered $u_j$ and $v_j$ has the vulnerability $q_j$. Each pair $(u_j, v_j)$ appears at most once in the input.

It is guaranteed that one or more highway networks that connect all the cities can be designed using some of these segments.

## Output

Output a line containing the minimum possible total risk severity.

### Sample Input 1

```
3 3
1 2 3
1 2 2
2 3 3
1 3 4
```

### Sample Output 1

```
16
```

### Sample Input 2

```
5 7
2 6 7 7 10
1 5 8
1 4 6
3 4 9
2 3 6
2 4 7
1 3 4
4 5 4
```

### Sample Output 2

```
210
```

# Problem D
## Tree Generators
Time Limit: 2 seconds

One of the problems in the International Parsing Contest caught your attention.

Two expressions are given as input, each representing a procedure to generate a single tree. The generation procedure is randomized, meaning different trees may be generated each time the procedure is performed. You are asked to count the number of trees that can possibly be generated from *both* of the expressions.

The syntax of an expression is as follows.

$$E \ ::= \ \text{`}\mathtt{1}\text{'} \ | \ \text{`}\mathtt{(}\text{'} \ E \ E \ \text{`}\mathtt{)}\text{'}$$

A tree is generated from an expression according to the following procedure.

- The expression `1` generates a tree with a single vertex labeled 1.
- For two expressions $E_1$ and $E_2$, an expression `(`$E_1E_2$`)` generates a tree as follows:
  - A tree $T_1$ is generated from $E_1$ with $n_1$ vertices, and $T_2$ from $E_2$ with $n_2$ vertices.
  - Then, the labels of all vertices in $T_2$ are incremented by $n_1$.
  - After that, two vertices, one from $T_1$ and the other from $T_2$, are randomly chosen. Adding an edge connecting them forms a single tree with vertices labeled 1 through $(n_1 + n_2)$, which is the tree generated by `(`$E_1E_2$`)`.

For example, the expression `(11)` can generate only the leftmost tree in Figure D.1, while `(1(11))` can generate the remaining two trees.

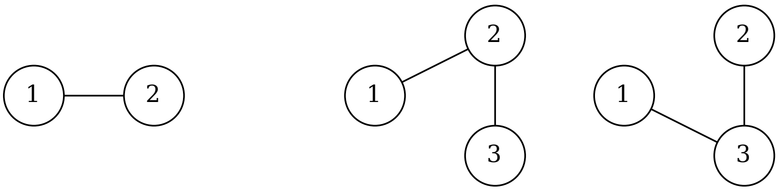

Figure D.1. Trees generated from the two expressions, `(11)` and `(1(11))`

The same tree may be generated from different expressions. The middle tree can also be generated from `((11)1)`.

For given two expressions of the same length, count the number of trees that can be generated from both of the expressions. Note that the trees generated from them always have the same number of vertices. Two trees are considered different if there exist two indices $i$ and $j$ such that vertices labeled $i$ and $j$ are connected by an edge in one tree but not in the other.

## Input

The input consists of two lines, each containing an expression string. The two strings have the same length, between $1$ and $7 \times 10^5$, inclusive, and follow the syntax given above.

## Output

Output the number of trees that can be generated from both expressions modulo $998\,244\,353$.

### Sample Input 1

```
((1(11))1)
((11)(11))
```

### Sample Output 1

```
1
```

### Sample Input 2

```
(1(11))
(1(11))
```

### Sample Output 2

```
2
```

### Sample Input 3

```
(((11)(11))((11)1))
((1(11))(1(1(11))))
```

### Sample Output 3

```
3
```

### Sample Input 4

```
((11)(((1(11))1)((11)1)))
(1(((11)((11)(11)))(11)))
```

### Sample Output 4

```
4
```

For Sample Input 1, the trees that can be generated from the two expressions are shown in Figure D.2. The top six trees correspond to the first expression and the bottom four correspond to the second. Only the leftmost tree in each row can be generated from both.

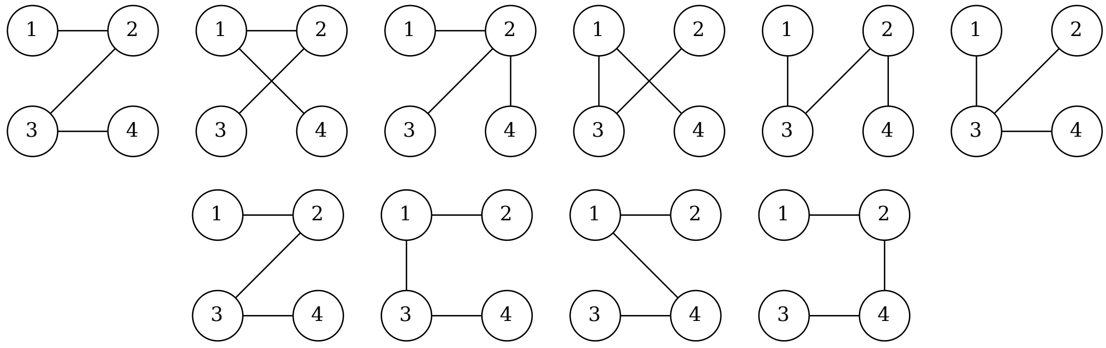

Figure D.2. Illustration of Sample Input 1

# Problem E
## E-Circuit Is Now on Sale!
Time Limit: 2 seconds

Are you looking for math education tools for your children? Then, why not try this amazing product, *E-circuit?* This is the best toy to learn two-dimensional geometry, logic, and arithmetic!

The E-circuit consists of a grid space with a bunch of square blocks, called *units*. Each unit perfectly fits into a grid cell. They have some input and/or output terminals to transfer integer values. When a number of units are *appropriately* placed in the grid, they form a tree representing a mathematical formula.

The units have different functionalities, each represented by a single character as follows.

- Digit (`0` to `9`): These units have one output terminal. Each of them sends the integer value indicated by the digit to its output.
- Connector (`#`): These units have one input terminal and one output terminal. Each of them receives an integer value from its input and sends the value to its output without making any changes.
- Operator (`+`, `-`, `*`, `/`): These units have two input terminals and one output terminal, do the following calculations on the values received from their inputs, and send the results to their outputs.
  - `+` operators compute the sum of the two input values.
  - `-` operators compute the difference of the two input values, the larger one subtracted by the smaller one.
  - `*` operators compute the product of the two input values.
  - `/` operators compute the quotient of the input values, the larger one divided by the smaller one, truncating the fraction, if any.
- Printer (`P`): The printer has one input terminal and displays the input value. There should be exactly one printer unit on the grid.

Two cells are adjacent if they share an edge. When two units are placed on adjacent cells, they are connected by using an input terminal of one unit and an output terminal of the other.

You are given an appropriate placement of units in which the units form a single tree representing a mathematical formula. A formal description of such a placement will be given in the Input section.

Your task is to calculate the value the printer displays for the given unit placement.

### Input

The input consists of a single test case of the following format.

$$\begin{array}{l} n\ m \\ x_{1,1}\ \cdots\ x_{1,m} \\ \vdots \\ x_{n,1}\ \cdots\ x_{n,m} \end{array}$$

The first two integers $n$ and $m$ $(1 \leq n \leq 50,\ 1 \leq m \leq 50)$ mean that the grid has cells arranged in an $n \times m$ matrix. The following $n$ lines describe the placement of units. The character $x_{i,j}$ $(1 \leq i \leq n,$ $1 \leq j \leq m)$ specifies the unit on the cell $i$-th from the top and $j$-th from the left. Each character either represents the unit functionality, as described in the problem statement, or is the character '`.`', meaning that the cell is empty.

It is guaranteed that the units are placed appropriately, that is,

- the number of adjacent units of each unit equals the total number of its input and output terminals,
- the total number of input terminals and that of output terminals of all the units are equal, and
- all the units belong to the printer tree: a unit belongs to the printer tree if and only if it is the printer or adjacent to another unit belonging to the printer tree.

It is also guaranteed that input values of '`/`' operators are not zero and no units have an output value larger than $10^{18}$.

## Output

Output a line containing the displayed value.

### Sample Input 1

```
6 8
3.......
#....P..
#....#.2
#.###*#+
##-....#
..1...4#
```

### Sample Output 1

```
12
```

### Sample Input 2

```
4 3
.4.
./P
9*.
.#7
```

### Sample Output 2

```
15
```

### Sample Input 3

```
5 11
8...8.....8
#.###...###
#.#...P.#..
#**##-*+/##
.4...3.0..1
```

### Sample Output 3

```
2024
```

# Problem F

## The Farthest Point

Time Limit: 2 seconds

An ant is on one of the vertices, say the *starting* vertex, of a rectangular cuboid (a hexahedron with all of its faces being rectangular). The surface of the cuboid constitutes the entire world of the ant.

We'd like to know which point on the surface of the cuboid is the farthest for the ant from the starting vertex. You may think that the *opposite* vertex, that is, the opposite end of the interior diagonal from the starting vertex, is the farthest. The opposite vertex is, however, not necessarily the farthest.

For example, on the surface of a cuboid of size $1 \times 1 \times 2$, the distance from a vertex to the opposite vertex is the square root of $8$. The distance to the farthest point is, however, the square root of $65/8$ (Figure F.1).

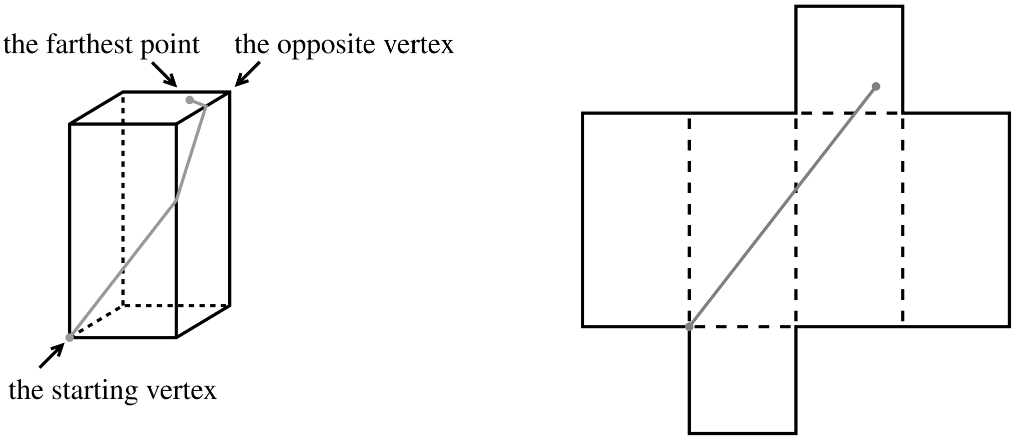

Figure F.1. Rectangular cuboid of size $1 \times 1 \times 2$, and its net

You are given the size of the rectangular cuboid. Write a program which calculates the distance from the starting vertex to the farthest point.

## Input

The input consists of a single test case of the following format.

$a$ $b$ $c$

The three integers $a$, $b$, and $c$ mean that the size of the rectangular cuboid is $a \times b \times c$. All of them are between 1 and 100, inclusive.

## Output

Output a line containing the distance from the starting vertex to the farthest point. The relative error of the output must be less than or equal to $10^{-9}$.

| Sample Input 1 | Sample Output 1 |
|---|---|
| `1 1 2` | `2.850438562747845` |

### Sample Input 2

```
10 10 10
```

### Sample Output 2

```
22.360679774997898
```

### Sample Input 3

```
100 2 3
```

### Sample Output 3

```
101.0503923792481
```

### Sample Input 4

```
2 3 5
```

### Sample Output 4

```
7.093659140387279
```

### Sample Input 5

```
84 41 51
```

### Sample Output 5

```
124.582755157578
```

# Problem G

## Beyond the Former Explorer

Time Limit: 4 seconds

You are standing at the very center of a field, which is divided into a grid of cells running north-south and east-west. A great treasure is hidden somewhere within one of these cells.

*John Belzoni*, a descendant of the famous treasure hunter *Giovanni Battista Belzoni,* actually discovered that treasure. Unfortunately enough, he died of heatstroke before successfully digging it out; he seems to have spent too long time wandering around the field.

John started his exploration from the central cell of the field where you stand now. All of his footprints leading to the treasure are left on the field, but you cannot identify the footprint on a cell until you reach there. The footprint on a cell indicates which of the four adjacent grid cells he proceeded to. It is known that John did not visit the same grid cell twice or more. You see a footprint on the central cell indicating that **John's first step was northward.**

There is exactly one treasure cell in the field, and you will recognize it only when you are on it.

A possible situation of the field is shown in Figure G.1. John's footprints are depicted as arrows in the cells. The treasure cell is depicted as '`G`'. The shaded cell is where you initially stand.

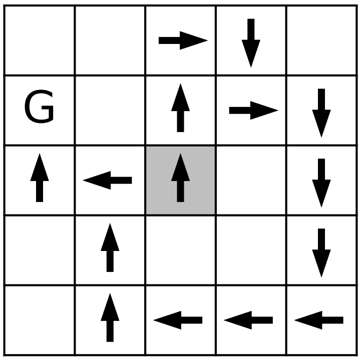

Figure G.1. Possible situation

Your task is to find the treasure in a limited number of *steps.* In a single step, you decide to take one of the four directions north, west, south, or east, and proceed to the adjacent cell in that direction. When you move on to the cell, you may find either the treasure, John's footprint, or nothing. You do not have to follow John's footprints. Unlike John's route, you may visit the same cell more than once. John's footprints will remain the same regardless of your exploration.

## Interaction

The interaction begins by receiving an integer $n$ $(1 \leq n \leq 2000)$ from the standard input, followed by a newline. The integer $n$ means that the field is partitioned into $(2n + 1) \times (2n + 1)$ grid cells. You are initially on the cell $(n + 1)$-th from the west end and $(n + 1)$-th from the north end. After receiving the integer $n$, you can start your exploration steps.

In each step, you send a single character meaning the direction of the cell to move to: '`^`' (caret) for north, '`<`' (less than symbol) for west, '`v`' (lowercase V) for south, and '`>`' (greater than symbol) for east. The character should be sent to the standard output followed by a newline.

In its response, you will receive a character indicating what you find on the cell you moved to, followed by a newline. The character ‘G’ means that the treasure is there. The characters ‘^’, ‘<’, ‘v’, and ‘>’ mean John’s footprint directing north, west, south, and east, respectively. The character ‘.’ (dot) means neither the treasure nor a footprint is on the cell.

When you find the treasure, that is, when you have received the character ‘G’, the interaction stops and your program should terminate. **You must reach the treasure cell within 30000 steps.** Although following John’s steps will surely lead you to the treasure, that may require more than 30000 steps.

In any of the following cases, your submission will be judged as a wrong answer.

- The output format of your program is invalid.
- You specify a direction to move out of the grid.
- Extra outputs are made after reaching the treasure cell.
- You could not reach the treasure cell within 30000 steps.

The arrangement of the field (the place of the treasure cell and John’s footprints) is fixed before the interaction starts; it does not change during the interaction.

As some environments require flushing the output buffers, make sure that your outputs are actually sent. Otherwise, your outputs will never reach the judge. You are provided with a command-line tool for local testing. For more details, refer to the clarification in the contest system.

### Sample Interaction

| Read | Write |
|---|---|
| 2 | |
| | ^ |
| ^ | |
| | < |
| . | |
| | v |
| < | |
| | < |
| ^ | |
| | ^ |
| G | |

In this interaction, the situation of the field in Figure G.1 is assumed.

# Problem H
## Remodeling the Dungeon 2
Time Limit: 8 seconds

The Queen of the Kingdom of *Icpca* resides in a castle peacefully. One day, she decided to remodel the dungeon of the castle.

The dungeon is a rectangular grid consisting of square cells. Some cells are enterable rooms, while others are duct spaces which are not enterable. All pairs of adjacent cells are separated by a wall. Some of the walls between two adjacent rooms are installed with a door to go back and forth. Any pair of rooms in the dungeon has connecting paths through these doors.

The Queen wants to remodel the dungeon so that there is only one unique path between any pair of rooms. Additionally, any pair of rooms both with only one door should be connected with a path going through an even number of doors. Due to the cost limitation, what can be done in the remodeling is only to block some (possibly zero) doors.

Your mission is to find a way to remodel the dungeon as the Queen wants.

## Input

The input consists of a single test case in the following format.

$$
\begin{array}{l}
h\ w \\
c_{1,1}c_{1,2}\cdots c_{1,2w+1} \\
c_{2,1}c_{2,2}\cdots c_{2,2w+1} \\
\vdots \\
c_{2h+1,1}c_{2h+1,2}\cdots c_{2h+1,2w+1}
\end{array}
$$

Two integers $h$ and $w$ mean that the dungeon size is $h \times w$. They are between $1$ and $400$, inclusive.

Each of the characters $c_{i,j}$ $(1 \le i \le 2h+1,\ 1 \le j \le 2w+1)$ is either '`.`' or '`#`'. These characters represent the configuration of the dungeon as follows.

- When both $i$ and $j$ are even, $c_{i,j}$ represents the cell located at the $i/2$-th row from the north and the $j/2$-th column from the west of the dungeon, referred to as cell $(i/2, j/2)$. Being '`.`' indicates that the cell is a room, while '`#`' indicates a duct space.
- When $i$ is odd and $j$ is even, it represents a wall. When $i = 1$ or $i = 2h+1$, it is a part of the outer wall of the dungeon. In this case, $c_{i,j}$ is always '`#`'. Otherwise, $c_{i,j}$ represents the wall between cells $((i-1)/2, j/2)$ and $((i+1)/2, j/2)$. Being '`.`' indicates that the wall has a door, while '`#`' indicates that it does not. Doors are installed only in walls between two rooms.
- When $i$ is even and $j$ is odd, it also represents a wall. When $j = 1$ or $j = 2w+1$, it is a part of the outer wall of the dungeon. In this case, $c_{i,j}$ is always '`#`'. Otherwise, $c_{i,j}$ represents the wall between cells $(i/2, (j-1)/2)$ and $(i/2, (j+1)/2)$. Being '`.`' indicates that the wall has a door, while '`#`' indicates that it does not. Doors are installed only in walls between two rooms.
- When both $i$ and $j$ are odd, $c_{i,j}$ is always '`#`', corresponding to an intersection of walls.

It is guaranteed that there is at least one room in the dungeon and any pair of rooms in the dungeon has one or more connecting paths.

## Output

If it is impossible to remodel the dungeon as the Queen wants, output `No`. Otherwise, output `Yes` on the first line, followed by the configuration of the dungeon after the remodeling in the same format as the input. If there are multiple possible configurations, any one of them is acceptable.

### Sample Input 1

```
3 3
#######
#.....#
#.#.###
#.#...#
#.#.#.#
#.....#
#######
```

### Sample Output 1

```
Yes
#######
#.....#
#.#####
#.#...#
#.###.#
#.....#
#######
```

### Sample Input 2

```
3 3
#######
#.....#
###.###
###...#
###.#.#
#.....#
#######
```

### Sample Output 2

```
Yes
#######
#.....#
###.###
###...#
#####.#
#.....#
#######
```

### Sample Input 3

```
3 3
#######
#.....#
#.###.#
#.###.#
#.###.#
#.....#
#######
```

### Sample Output 3

```
No
```

# Problem I
## Greatest of the Greatest Common Divisors
Time Limit: 4 seconds

You are given a sequence of integers and a number of intervals in the sequence. The intervals are specified by their leftmost and rightmost positions. An interval consisting of $k$ integers has $k(k-1)/2$ pairs of integers at different positions, which have their greatest common divisors. For each given interval, find the greatest one among such greatest common divisors.

For example, when the sequence is $(a_1, \ldots, a_6) = (10, 20, 30, 40, 50, 60)$, and the whole sequence is specified as the interval, the following $15$ pairs of two integers at different positions $x$ and $y$, and their greatest common divisors should be considered.

| $x$ | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | **3** | 4 | 4 | 5 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $y$ | 2 | 3 | 4 | 5 | 6 | 3 | 4 | 5 | 6 | 4 | 5 | **6** | 5 | 6 | 6 |
| $a_x$ | 10 | 10 | 10 | 10 | 10 | 20 | 20 | 20 | 20 | 30 | 30 | **30** | 40 | 40 | 50 |
| $a_y$ | 20 | 30 | 40 | 50 | 60 | 30 | 40 | 50 | 60 | 40 | 50 | **60** | 50 | 60 | 60 |
| $\gcd(a_x, a_y)$ | 10 | 10 | 10 | 10 | 10 | 10 | 20 | 10 | 20 | 10 | 10 | **30** | 10 | 20 | 10 |

The greatest of the greatest common divisors of the $15$ pairs is $\gcd(30, 60) = 30$, in this case.

### Input

The input consists of a single test case of the following format.

$$
\begin{array}{l}
n \\
a_1 \cdots a_n \\
q \\
l_1 \; r_1 \\
\vdots \\
l_q \; r_q
\end{array}
$$

The first line contains an integer $n$, which is the number of integers in the given sequence, satisfying $2 \leq n \leq 10^5$. The second line contains $n$ positive integers $a_1$ through $a_n$, specifying the sequence. None of them exceeds $10^5$.

The third line contains a positive integer $q$, specifying the number of intervals in the sequence to be considered, which does not exceed $10^5$. It is followed by $q$ lines, each specifying an interval in the sequence to be considered. The $i$-th line of them has two integers, $l_i$ and $r_i$ ($1 \leq l_i < r_i \leq n$), specifying the interval $a_{l_i}$ through $a_{r_i}$ in the sequence.

### Output

Output $q$ lines, the $i$-th line of which should have the greatest of the greatest common divisors of all pairs in the interval specified by $l_i$ and $r_i$.

### Sample Input 1

```
6
10 20 30 40 50 60
3
1 6
2 5
4 5
```

### Sample Output 1

```
30
20
10
```

### Sample Input 2

```
10
13 2 35 4 13 2 5 1 7 4
5
1 4
4 10
3 8
3 9
1 10
```

### Sample Output 2

```
2
4
5
7
13
```

# Problem J
## Mixing Solutions
Time Limit: 2 seconds

Let's prepare for an experiment with the chemical *Yokohama Yellow,* or *YY* in short. You have several containers of aqueous solution of YY. While YY is evenly dissolved in each solution, the concentration may differ among containers. You will take arbitrary amounts of solution from some of the containers and mix them to prepare a new solution with the predetermined total amount.

Ideally, the mixed solution should contain the *target amount* of YY, but there is a problem. While the exact amount of solution in each container is known, the amount of YY in each solution is guaranteed only to fall within a certain range. Due to this uncertainty, it is difficult to match the amount of YY in the mixed solution exactly to the target amount. Still, you can ensure that the error, the difference from the target amount, will never exceed a certain limit.

To be more precise, let the target and actual amounts of YY in the mixed solution be $y_\mathrm{t}$ mg (milligrams) and $y_\mathrm{a}$ mg, respectively. Given the amounts of solution taken from the containers, $y_\mathrm{a}$ is guaranteed to fall within a certain range. The *maximum error* is defined as the maximum of $|y_\mathrm{a} - y_\mathrm{t}|$ when $y_\mathrm{a}$ varies within this range.

Find the minimum achievable value of the maximum error, given that you can take any portion of the solution in each container as long as their total is equal to the predetermined amount.

### Input

The input consists of a single test case of the following format.

$$
\begin{array}{l}
n\ s\ c \\
a_1\ l_1\ r_1 \\
\vdots \\
a_n\ l_n\ r_n
\end{array}
$$

The first line contains three integers, $n$, $s$, and $c$, satisfying $1 \le n \le 1000$, $1 \le s \le 10^5$, and $0 \le c \le M$, where $M = 10^4$ here and in what follows. Here, $n$ denotes the number of containers of YY solution. The predetermined total amount of the mixed solution is $s$ mg, and the target amount of YY is $\frac{c}{M}s$ mg. The $i$-th of the following $n$ lines contains three integers, $a_i$, $l_i$, and $r_i$, satisfying $1 \le a_i \le 10^5$ and $0 \le l_i \le r_i \le M$. These integers indicate that the $i$-th container has $a_i$ mg of solution and that the amount of YY in it is guaranteed to be between $\frac{l_i}{M}a_i$ mg and $\frac{r_i}{M}a_i$ mg, inclusive. They satisfy $\sum_{i=1}^{n} a_i \ge s$.

### Output

The minimum achievable value of the maximum error can be proven to be a rational number. Express the value as an irreducible fraction $p/q$ with $q > 0$, and output $p$ and $q$ separated by a space on a single line.

### Sample Input 1

```
3 10 5000
10 2000 3000
10 4000 6000
10 7000 8000
```

### Sample Output 1

```
1 2
```

### Sample Input 2

```
2 10 5000
7 4500 5500
12 3500 6000
```

### Sample Output 2

```
4 5
```

### Sample Input 3

```
3 1 4159
1 1 1
1 100 100
1 10000 10000
```

### Sample Output 3

```
0 1
```

### Sample Input 4

```
6 12345 6789
2718 2818 2845
9045 2353 6028
7471 3526 6249
7757 2470 9369
9959 5749 6696
7627 7240 7663
```

### Sample Output 4

```
23901191037 67820000
```

# Problem K

## Scheduling Two Meetings

Time Limit: 1 second

You are the chief judge of the International Collegiate Quiz Contest (ICQC) this year. You want to hold judge meetings twice for preparing the problem set of the contest. You proposed candidate schedules for the meetings, and all the judges answered for each of the time slots whether they would attend the meeting on-site or remotely via a video conference tool.

You have to choose a pair of two distinct time slots, so that every judge attends at least one of the two meetings on-site. When there are multiple such pairs, you'd like to choose one with the largest number of judges attending both meetings on-site. If multiple pairs are still equally desirable with these criteria, one with the earlier first meeting is preferred. If there still remain multiple pairs with the same time slot for the first meeting, the one with the earliest second meeting should be chosen.

### Input

The input consists of a single test case of the following format.

$$\begin{array}{l} n\ m \\ a_{1,1}\ \cdots\ a_{1,m} \\ \vdots \\ a_{n,1}\ \cdots\ a_{n,m} \end{array}$$

Two integers $n$ and $m$ are given in the first line. The first integer $n$ ($2 \leq n \leq 2 \times 10^5$) is the number of candidate time slots. Here, the candidate time slots are numbered $1$ through $n$, and smaller numbers mean earlier time slots. The second integer $m$ ($2 \leq m \leq 20$) is the number of judges.

In the following $n$ lines, $a_{i,j}$ is either a character `Y` indicating that the $j$-th judge attends a meeting at the $i$-th candidate time slot on-site, or a character `N` indicating remote attendance.

### Output

Output a line containing the two time slot numbers of the most preferable choice, separated by a space, with the earlier time slot first. If there are no pairs of time slots satisfying the condition, output `No`.

| Sample Input 1 | Sample Output 1 |
|---|---|
| 4 3<br>NNY<br>YYN<br>YNY<br>NYY | 2 3 |

| Sample Input 2 | Sample Output 2 |
|---|---|
| 3 6<br>NNNYYY<br>YYNYYN<br>YYYNNN | 1 3 |

**Sample Input 3**

```
6 5
NNNNN
YNNNY
YYNNN
YYNNN
NYYNY
NNYYY
```

**Sample Output 3**

```
3 6
```

**Sample Input 4**

```
3 3
YNN
NYN
NNY
```

**Sample Output 4**

```
No
```

**Sample Input 5**

```
4 4
NYNN
YNYY
YNYN
NNYY
```

**Sample Output 5**

```
1 2
```

# Problem L

## Peculiar Protocol

Time Limit: 2 seconds

The Kingdom of *Icpca* has a peculiar protocol in wedding ceremonies. That is, amounts of monetary gifts should be a multiple of a fixed quantity plus a fixed extra. When the fixed quantity is $d$ and the fixed extra is $r$, courteous amounts of wedding gifts are $k \times d + r$ for any non-negative integer multipliers $k$.

Initially, you have a pile of $n$ banknotes. Every time you attend a wedding ceremony, you draw out a contiguous portion from your current banknote pile as your gift that sums up to a courteous amount, that is, a multiple of $d$ plus additional $r$. If no contiguous portion sums up to such an amount, you cannot attend wedding ceremonies any more. After drawing out, the remaining banknotes are squeezed to form a single pile, keeping their relative order. The resultant pile of banknotes may have portions with such an amount, which allows you to attend more ceremonies.

Your monetary gifts are expected to raise your social reputation. As the additional amount $r$ is considered mandatory, the multiplier $k$ is considered significant. Your reputation is raised in proportion to $k$ at each of the ceremonies you attend.

For example, assume $d = 5$ and $r = 1$, and you have banknotes whose values are 2, 2, 2, 4, and 4, piled up in this order. When you attend a wedding ceremony, there are following two possible ways to give courteous monetary gifts.

- Give a monetary gift consisting of the first three banknotes from the top, totaling $2 + 2 + 2 = 6 = 1 \times d + r$. After drawing them out, you have banknotes with values 4 and 4. No contiguous portion of your remaining banknote pile sums up to a courteous amount. Thus, you cannot attend wedding ceremonies anymore.
- Give a monetary gift consisting of the third and the fourth banknotes, totaling $2+4 = 6 = 1 \times d+r$. After drawing them out, you have banknotes with values 2, 2, and 4, in this order. You can attend another wedding ceremony because the second and the third banknotes total $2+4 = 6 = 1 \times d+r$, which is courteous.

In this example, the second way can maximize your social reputation by attending two ceremonies, because the total of the multipliers is $1 + 1 = 2$, which achieves the maximum possible.

In contrast, if the value of the first banknote is 12, giving the first three banknotes at a ceremony prevents you from attending more ceremonies. That, however, maximizes your social reputation because the total of the multipliers is 3, which achieves the maximum possible.

Compute the maximum possible total of the multipliers with your monetary gifts at wedding ceremonies. You may assume that you have so many unmarried relatives and friends that you can attend any number of wedding ceremonies as long as you can give courteous monetary gifts.

## Input

The input consists of a single test case in the following format.

$$\begin{array}{l} n\ d\ r \\ a_1 \cdots a_n \end{array}$$

The first line has three integers $n$, $d$, and $r$. The integer $n$ ($1 \le n \le 500$) is the number of banknotes you have. The integers $d$ and $r$ ($2 \le d \le 10^8$, $0 \le r < d$) represent the parameters of the peculiar protocol. The second line has $n$ integers, $a_1, \ldots, a_n$. Here, $a_i$ ($0 \le a_i \le 10^8$) represents the value of the banknote $i$-th from the top.

## Output

Output a line containing the maximum possible total of the multipliers with your monetary gifts at wedding ceremonies.

### Sample Input 1

```
5 5 1
2 2 2 4 4
```

### Sample Output 1

```
2
```

### Sample Input 2

```
5 5 1
12 2 2 4 4
```

### Sample Output 2

```
3
```

### Sample Input 3

```
5 20000 10000
5000 10000 15000 5000 25000
```

### Sample Output 3

```
2
```

### Sample Input 4

```
9 5 3
4 2 2 1 1 4 3 2 1
```

### Sample Output 4

```
2
```