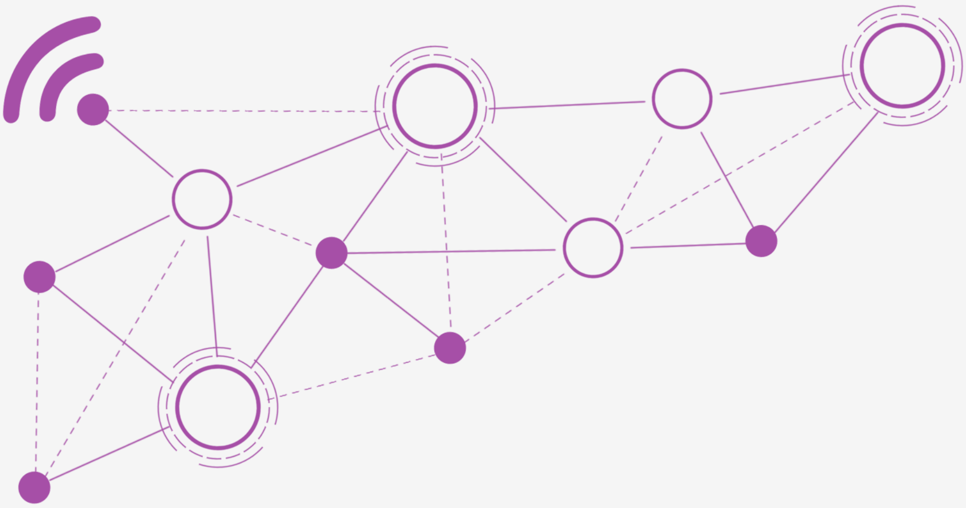


noi.ph

2022

# National Olympiad in Informatics

## Elimination Round

# Contents

# Notes

- Many problems have large input file sizes, so use fast I/O. For example:
  - In Java, use `BufferedReader` and `PrintWriter`.
  - In C/C++, use `scanf` and `printf`.
- Good luck and enjoy the problems!

## **Important!** Read the following:

- Your solution will be checked by running it against one or more (usually several) hidden test cases. You will not have access to these cases, but a correct solution is expected to handle them correctly.
- The output checker is **strict**. Follow these guidelines strictly:
  - It is **space sensitive**. Do not output extra leading or trailing spaces. Do not output extra blank lines unless explicitly stated.
  - It is **case sensitive**. So, for example, if the problem asks for the output in lowercase, follow it.
  - Do not print any tabs. (No tabs will be required in the output.)
  - Do not output anything else aside from what's asked for in the Output section. So, do not print things like "Please enter t".

  Not following the output format strictly and exactly will likely result in the verdict "*Output isn't correct*".
- Do not read from, or write to, a file. You must read from the standard input and write to the standard output.
- For Java, do not add a package line at the top of your code. Otherwise, your submission will receive the verdict "*Execution failed because the return code was nonzero*" or "*Compilation failed*".
- Only include one file when submitting: the source code (.cpp, .java, .py, etc.) and nothing else.
- Only use letters, digits and underscores in your filename. Do not use spaces or other special symbols.
- Many problems have large input file sizes, so use fast I/O. For example:
  - In Java, use `BufferedReader` and `PrintWriter`.
  - In C/C++, use `scanf` and `printf`.

  We recommend learning and using these functions during the Practice Session.
- Good luck and enjoy the contest!

# Problem A
## Bruno Gets Disqualified

Time Limit: 1 second

Last year, our dear friend Bruno got disqualified from NOI.PH. He broke a rule and was of course disqualified. He sent an appeal but the reply he got was a big fat No... No... No...!

This year, Bruno broke $x$ rules that he knew about, and broke $y$ rules that he did not know about. Will he be disqualified?

If you need help figuring out the answer, feel free to look at `https://noi.ph/rules`

## Input Format

Input consists of a single line with two space-separated non-negative integers, $x$ and $y$, whose meanings were described in the problem statement.

## Output Format

Output a single line, containing the verdict for Bruno.

- If Bruno is not disqualified, output `BruNoNoNo`
- If Bruno is disqualified, output `BruYesYesYes`

## Constraints and Subtasks

| For all subtasks |
|---|
| $0 \leq x, y \leq 9$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **33** | It is guaranteed that $x = 0$ |
| 2 | **33** | It is guaranteed that $y = 0$. |
| 3 | **34** | No further constraints. |

### What does "Constraints" mean?

"Constraints" means, "Your program will only be tested on input data where the following hold."

For example, in this problem, since the constraints say that $0 \leq x, y \leq 9$, your program does **not** need to handle inputs where either number of violations is less than 0 or greater than 9.

Your program does **not** need to check if the input data is valid and report errors (or something) if the constraints are not followed. The "Constraints" are our promise that we will not judge your submission using inputs that do not follow the constraints.

Different subtasks have different constraints. Different constraints change the difficulty of the problem. Easier subtasks require your program to handle fewer possible test cases. For example, in this problem, if you are only targetting subtask 1 (for 33 points), your program only needs to handle inputs where $x = 0$. If your program works properly on test cases that follow this constraint, but fails on cases where $x \neq 0$, then you still get 33 points.

## Sample I/O

| Input 1 | Output 1 |
|---|---|
| `0 0` | `BruNoNoNo` |

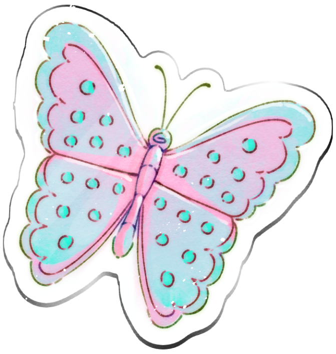
| Input 2 | Output 2 |
|---|---|
| `1 1` | `BruYesYesYes` |

| Input 3 | Output 3 |
|---|---|
| `0 1` | `BruYesYesYes` |

# Problem B

## Paru-Paro G

Time Limit: 1 second



One fateful afternoon, Maypie Netta Pacia found herself chasing a field butterfly for fun. While doing so, she accidentally bumped into someone who would eventually be her fiancé—Juan Uy.

For their wedding, they call their entourage the Paro Paro Group—Paro Paro G for short. This is because they planned to have each member of their entourage have a butterfly icon embossed on their outfit, in honor of the butterfly that brought them together! The size of the outfit, and hence the butterfly icon to be embossed on it, varies depending on the size of the person it fits. Therefore, they would need different sizes for this butterfly image.

Your task is to help the future Mrs. Maypie Netta Pacia-Uy by giving her samples of the butterfly logo in different sizes. Sizes are given in terms of an integer $n$.

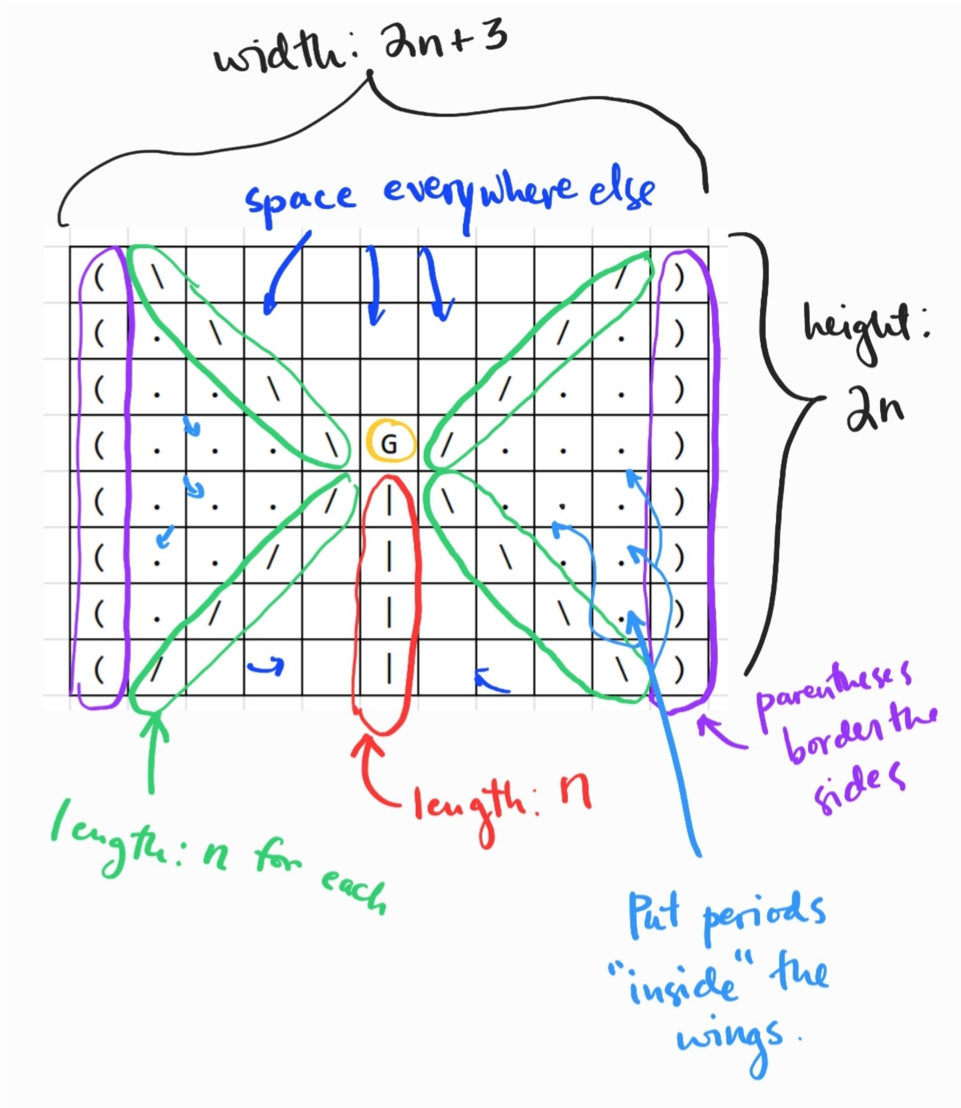
Here is the butterfly logo of size $n = 2$:

```
(\   /)
(.\G/.)
(./|\.)
(/ | \)
```

Here is the butterfly logo of size $n = 4$:

```
(\       /)
(.\     /.)
(..\   /..)
(...\G/...)
(.../|\...)
(../ | \..)
(./  |  \.)
(/   |   \)
```

In general, here is an annotated blueprint detailing what the butterfly logo should look like, for general $n$. Note the whitespaces.



```
(\       /)
(.\     /.)
(..\   /..)
(...\G/...)
(.../|\...)
(../ | \..)
(./  |  \.)
(/   |   \)
```

## Input Format

Input consists of a single line containing a single integer $n$.

## Output Format

Output the butterfly logo of size $n$. As a reminder, this should consist of exactly $2n$ lines, each containing exactly $2n + 3$ characters.

This problem will be strict about trailing spaces and newlines, so be precise! Do not output more than is necessary.

## Constraints and Subtasks

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **49** | $2 \leq n \leq 5$ |
| 2 | **51** | $2 \leq n \leq 50$ |

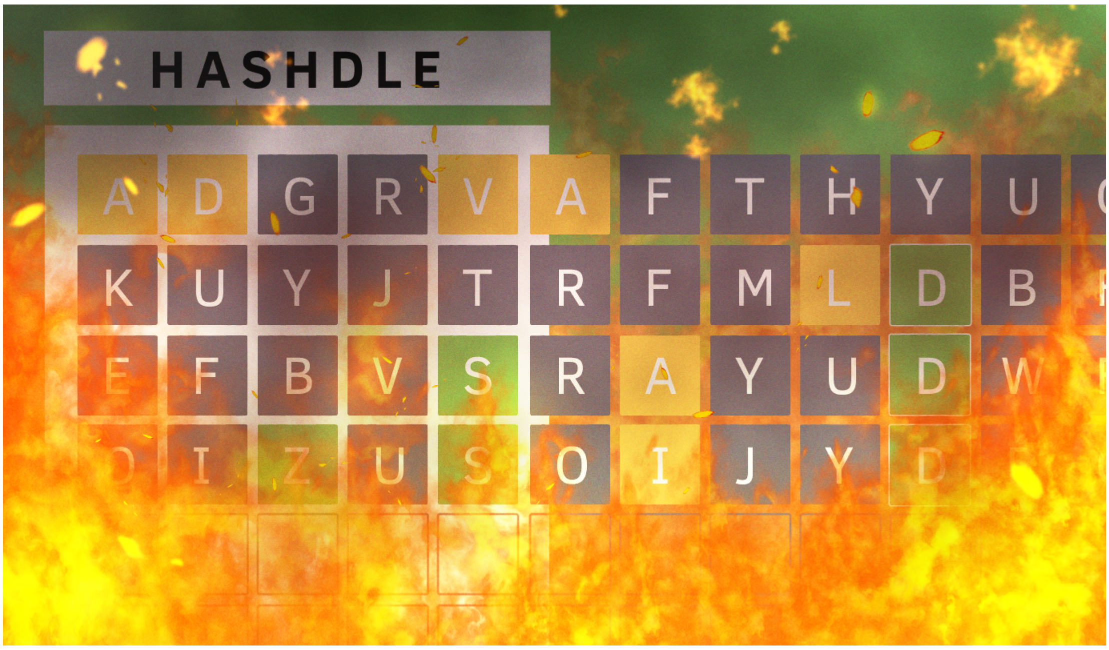
## Sample I/O

| Input 1 | Output 1 |
|---|---|
| `2` | `(\   /)`<br>`(.\G/.)`<br>`(./\|\.)`<br>`(/ \| \)` |

| Input 2 | Output 2 |
|---|---|
| `4` | `(\       /)`<br>`(.\     /.)`<br>`(..\   /..)`<br>`(...\G/...)`<br>`(.../\|\...)`<br>`(../ \| \..)`<br>`(./  \|  \.)`<br>`(/   \|   \)` |

# Problem C
## Cryptographic Hashdle

Time Limit: 1 second



A certain *Word Game That Shall Not Be Named* has taken the internet by storm, setting off a chain reaction of copycats and variants that reimagine the game in different languages, dimensions, and levels of absurdity. This is probably the best thing to have come out of 2022 so far.

As your contribution to this wonderful craze, you have to decided to create *Cryptographic Hashdle*, a game that's easy to learn but difficult to master, suitable for all ages!

The game is incredibly simple. You, the player, must determine the value of some hidden string $s$. Here, $s$ is some arbitrary string that may contain up to one million alphanumeric characters, and which likely has no semantic meaning in any existing language. Pretty straightforward, right? Sounds easy enough!

Guesses take the form of a string $t$, which is the same length as $s$. Each character in $t$ will be highlighted one of three colors—green, yellow, or grey—according to the following rules.

- If some character in $t$ matches the character at the same location in $s$, highlight it green.
- If some character in $t$ exists *somewhere* in $s$, but not at this location, highlight it yellow.
- If some character in $t$ does not appear at all in $s$, highlight it grey.

Cryptographic Hashdle allows for both upper and lowercase letters, and they are considered **distinct** from each other. In other words, this game is case sensitive!

Please read and understand these rules properly. They do not necessarily reflect the highlighting rules in the original *Word Game That Shall Not Be Named* or its variants, particularly in the case of duplicates and yellows. These exact rules given in the problem statement, and these rules only, will be the basis of correctness for this problem.

Given two strings $s$ and $t$, apply the above rules to determine what color to highlight each character of $t$.

## Input Format

The first line of input contains a single positive integer $n$, the length of the secret word, and of the guess.

The second line of input contains the secret string $s$.

The third line of input contains the guess $t$.

## Output Format

Output a single line containing a string of length $n$, according to the highlighting rules described above.

- Green characters should be represented by `G`
- Yellow characters should be represented by `Y`
- Grey characters should be represented by `.` (a period)

## Constraints and Subtasks

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **23** | $1 \le n \le 5$ |
| 2 | **23** | $1 \le n \le 100$ |
| 3 | **23** | $1 \le n \le 5000$ |
| 4 | **15** | $1 \le n \le 2 \times 10^5$ |
| 5 | **16** | $1 \le n \le 10^6$ |

## Sample I/O

| Input 1 | Output 1 |
|---|---|
| 5<br>NOI22<br>ORION | Y.GYY |

| Input 2 |
|---|
| 32<br>QuickBrownFoxJumpsOverTheLazyDog<br>PackMyBoxWithFiveDozenLiquorJugs |

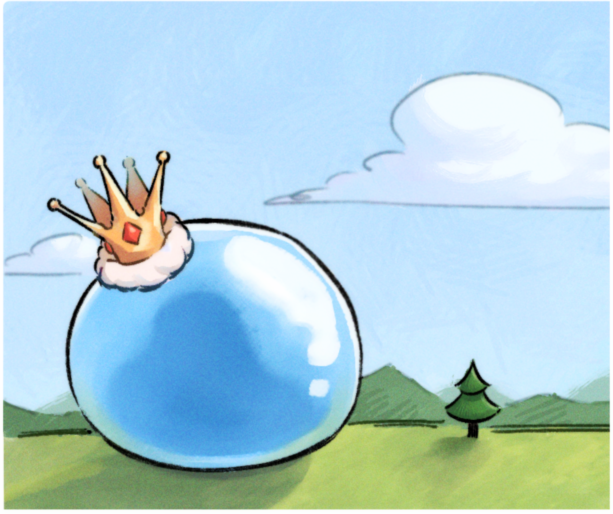
| Output 2 |
|---|
| .YYY.YYGY.Y.YYYYYYYYGYYY.YYYYYYY |

| Input 3 | Output 3 |
|---|---|
| 25<br>SHA256leActuallyExistsBTW<br>IsntThatJustBitcoinMining | .Y.YY.YY.YYYYYYY.Y..Y.Y.. |

| Input 4 | Output 4 |
|---|---|
| 8<br>1I1iIII1<br>11Iill1i | GYYG..YY |

# Problem D
## Slime King

Time Limit: 1 second



*There has always been... there can only be... there **will** only be...*

One... true...

**Slime King.**

Although Jay Arpeggio would say that he is but a humble breeder of monsters and creatures, any adventurer or villain worth their salt knows that he has been the premier supplier of goons and mooks to Evil Overlords and Tyrant Kings for close to three decades now.

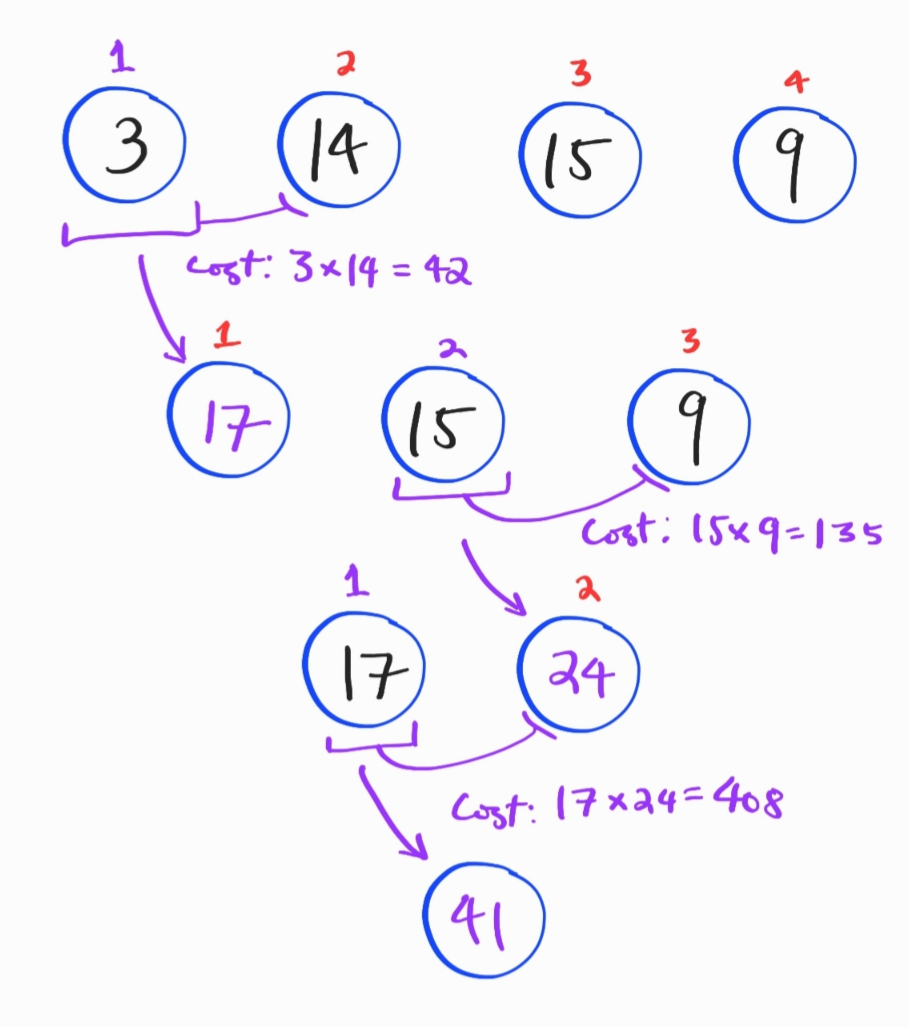
Jay is usually bored by bog-standard orders—goblins, kobolds, mandragora—but today, he got a fun order. He has been tasked with creating a **Slime King**.

Jay has $n$ slimes arranged in a row, whose sizes are $a_1, a_2, a_3, \ldots, a_n$, in that order. These slimes are not yet the Slime King! No, there can only be one true Slime King. Jay can choose any slime (except for the last one) and force it to **absorb** the slime that comes immediately after it in line. Formally, after he chooses some $i$, the following happen:

- A cost of $a_i \times a_{i+1}$ is incurred, due to waste emitted by the absorption process.
- The value of $a_i$ (the size of the $i$th slime) increases by $a_{i+1}$.
- $a_{i+1}$ is deleted (and the indices of the slimes that come after shift backwards to adjust)

Note that the relative order of the slimes never changes. After doing this $n - 1$ times, there will only be one slime remaining. **This is the Slime King**. The total cost to create a Slime King is just the sum of the costs of the individual absorption operations.

For example, suppose we have 4 slimes with sizes $\{3, 14, 15, 9\}$. Here is one possible series of operations, with a total cost of 585 to create the Slime King.



Normally, Jay would aim to find the minimum possible cost required to create a Slime King. But that's boring! Instead, Jay considers the following more interesting problem.

Suppose Jay selects the absorptions totally randomly, i.e. at the $k$th step, he uniformly randomly selects an index $i$ from the range 1 to $n - k$, and forces the $i$th slime to perform the absorption. On average, across all the possible absorption orders, what is the expected total cost required to create a Slime King?

## Input Format

The first line of input contains a single integer $n$, the number of slimes.

The second line of input contains $n$ space-separated integers $a_1, a_2, \ldots, a_n$, the initial sizes of the slimes.

## Output Format

Output a line containing a single real number, the average total cost to create a Slime King across all possible absorption orders.

Your answer will be accepted if it has an absolute error of at most $10^{-12}$ from the judge's answer. In symbols, let $a$ be your answer and let $b$ be the judge's answer—your answer will be accepted if $|a - b| < 10^{-12}$. Informally, your answer should be correct to at least 12 decimal places.

## Constraints and Subtasks

| For all subtasks |
|---|
| $2 \le n \le 2 \times 10^5$<br>$1 \le a_i \le 12358$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **50** | $2 \le n \le 5$ |
| 2 | **12** | $2 \le n \le 100$ |
| 3 | **18** | $2 \le n \le 5000$ |
| 4 | **12** | $a_i = 1$ for all $i$. |
| 5 | **8** | No further constraints. |

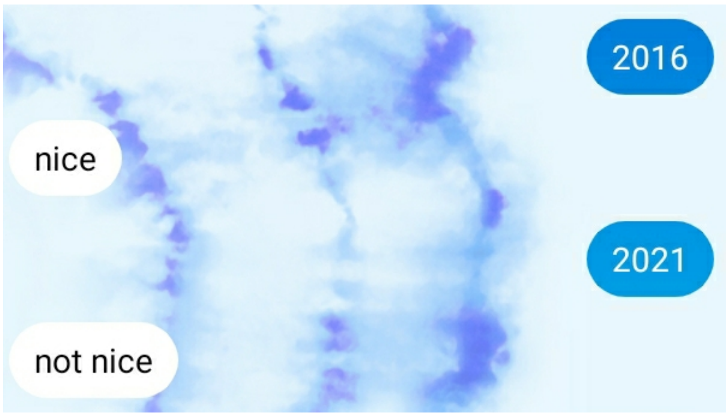
## Sample I/O

| Input 1 | Output 1 |
|---|---|
| `3`<br>`2 3 5` | `31.000000000000` |

| Input 2 | Output 2 |
|---|---|
| `10`<br>`1 1 2 3 5 8 13 21 34 55` | `7777.000000000000` |

# Problem E
## Nice Numbers (Decision)

Time Limit: 1 second



*Nice Numbers (Decision) and (Counting) begin with the same definition of nice numbers, and only differ in the task given in the final paragraph.*

Bro.

Bro.

I thought of something nice bro.

What is it bro?

2016, bro.

2016 is nice, bro?

2016 is definitely nice, bro.
You can write $2016 = 4 \times 7 \times 8 \times 9$, bro.

Woah, bro. That's like, super nice.
Mega satisfying, bro.

I think we can generalize this, bro.
I think any positive integer which can be expressed as the product of one digit numbers should be considered *nice*, bro.

Totally agree, bro. So like, 2025?

2025 is nice, bro.
$2025 = 3 \times 3 \times 5 \times 5 \times 9$.

And 2020 and 2021, bro?

*Not nice* at all bro. Super not nice.

I getchu bro. We should write a program that identifies years that are nice numbers, bro.

Think bigger, bro. How about nice *nanoseconds since epoch*?

I'm so in, bro.

But the bros aren't good at coding. Help them out, would you? Write a program that, given a positive integer $n$, determines if $n$ is nice.

## Input Format

The first line of input contains a single integer $T$, the number of test cases. Then, $T$ lines follow, each describing a test case in the following manner.

Each test case consists of a single line containing a single positive integer $n$, the number whose niceness is to be determined.

## Output Format

For each test case, output `nice` if the given number is nice, and `not nice` if it is not.

## Constraints and Subtasks

| **For all subtasks** |
|---|
| $1 \leq T \leq 10^5$<br>$1 \leq N \leq 10^{18}$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **20** | $1 \leq T \leq 10$<br>$1 \leq N \leq 100$ |
| 2 | **20** | $1 \leq T \leq 10$<br>$1 \leq N \leq 10^5$ |
| 3 | **20** | $1 \leq T \leq 10$<br>$1 \leq N \leq 10^9$ |
| 4 | **20** | $1 \leq N \leq 10^5$ |
| 5 | **20** | No further constraints. |

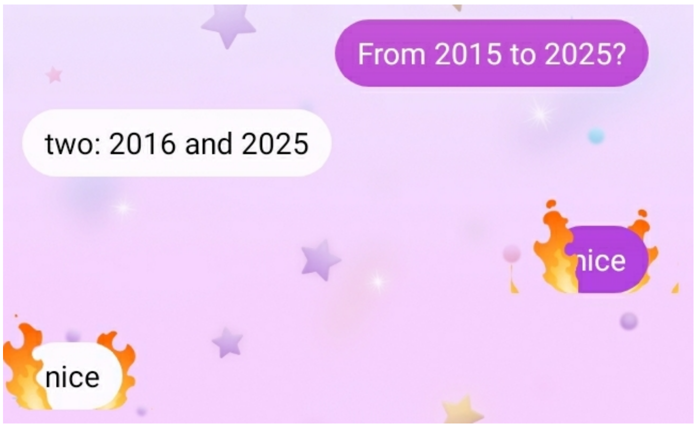
## Sample I/O

| Input 1 | Output 1 |
|---|---|
| 4<br>2016<br>2020<br>2021<br>2025 | nice<br>not nice<br>not nice<br>nice |

| Input 2 | Output 2 |
|---|---|
| 3<br>69<br>420<br>1000000000000000000 | not nice<br>nice<br>nice |

# Problem F
## Nice Numbers (Counting)

Time Limit: 1 second



From 2015 to 2025?

two: 2016 and 2025

nice

nice

*Nice Numbers (Decision) and (Counting) begin with the same definition of nice numbers, and only differ in the task given in the final paragraph.*

Bro.

Bro.

I thought of something nice bro.

What is it bro?

2016, bro.

2016 is nice, bro?

2016 is definitely nice, bro.
You can write $2016 = 4 \times 7 \times 8 \times 9$, bro.

Woah, bro. That's like, super nice.
Mega satisfying, bro.

I think we can generalize this, bro.
I think any positive integer which can be expressed as the product of one digit numbers should be considered *nice*, bro.

Totally agree, bro. So like, 2025?

2025 is nice, bro.
$2025 = 3 \times 3 \times 5 \times 5 \times 9$.

And 2020 and 2021, bro?

*Not nice* at all bro. Super not nice.

I getchu bro. We should write a program that **counts** years that are nice numbers, bro.

Think bigger, bro. How about nice *nanoseconds since epoch*?

I'm so in, bro.

But the bros aren't good at coding. Help them out, would you? Write a program that, given positive integers $L$ and $R$, **counts** the nice numbers between $L$ and $R$, inclusive. Formally, count the number of integers $x$ such that $L \leq x \leq R$ and $x$ is nice.

## Input Format

The first line of input contains a single integer $T$, the number of test cases. Then, $T$ lines follow, each describing a test case in the following manner.

Each test case consists of a single line containing two space-separated positive integers $L$ and $R$, the (inclusive) endpoints of the range we will be counting over.

## Output Format

For each test case, output a single integer, the number of nice numbers in the given range.

## Constraints and Subtasks

| **For all subtasks** |
|---|
| $1 \leq T \leq 10^5$<br>$1 \leq L \leq R \leq 10^{18}$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **20** | $1 \leq T \leq 10$<br>$1 \leq L \leq R \leq 100$ |
| 2 | **10** | $1 \leq T \leq 10$<br>$1 \leq L \leq R \leq 10^5$ |
| 3 | **15** | $1 \leq L \leq R \leq 10^5$ |
| 4 | **10** | $1 \leq T \leq 10$<br>$1 \leq L \leq R \leq 10^9$<br>$\lvert R - L \rvert \leq 100$ |
| 5 | **15** | $1 \leq T \leq 10$<br>$\lvert R - L \rvert \leq 100$ |
| 6 | **10** | $1 \leq T \leq 10$ |
| 7 | **20** | No further constraints. |

## Sample I/O

| Input 1 | Output 1 |
|---|---|
| `2`<br>`1 15`<br>`2015 2025` | `13`<br>`2` |

| Input 2 | Output 2 |
|---|---|
| `1`<br>`999999999999999999 1000000000000000000` | `1` |

## Explanation

For the first test case of the first sample input, we can show that 11 and 13 are not nice.

# Problem G
## Pampamilyang Pasalubong

Time Limit: 1.5 seconds

E. Gabriella Halohalo is lucidly managing national inventories and product lineups for her nationwide souvenir chain store, “Ngiting Only In the Philippines House”. Market research suggests that their offerings lack in rice desserts, so the NOIPH team has tried to come up with concepts to address this.

One such concept they’ve come with is the *Shiny Sapin-Sapin* (SSS, for short), a variation on a rice dessert already known to many Filipino homes. Their twist on it is the addition of not one, not two, but *seven* new flavors:

- Vivid Ube, an inclusion from the delicacy’s traditional flavors.
- Ooh-la-la Orange, a zesty twist sure to surprise customers.
- Gorgeous Green Grapes, a rich addition that will deepen the palette.
- Rad Rambutan, a wise touch to keep the flavors close to home.
- Yummy Langka, a sweet flavor that blends well with anything.
- Blissful Blueberries, a new combination that improves texture and flavor.
- White Chocolate, a modern infusion to the well-loved recipe. (Or so your pitch deck claims...)

NOIPH has already canvassed distributors for each of these 7 flavors, and already has the pipeline for preparation planned out. All they’ll have to do is to design the SSS itself.

Popular demand suggests than an SSS will make the most sales if it is $h$ centimeters long and $w$ centimeters wide. Also, for easy sharing, the SSS will be divided into $hw$ square pieces, each with a side length of 1 centimeter, such that each square piece is flavored with one of their 7 flavors.

However, a problem has arisen! Each of their seven suppliers has reported that they only manufacture SSS pieces in chunks of four one-by-one squares, with each supplier only providing chunks in some particular fixed shape. See the Explanation section for a full description of all the shapes.

Moreover, when being used to assemble an SSS, they have provided the following guidelines:

- The chunk **should not** be flipped over; as otherwise, the flavoring will stick to the bottom of the packaging, which makes cleanup very messy. The pieces *may* be rotated, though.
- The chunk **should not** be placed beside chunks with the same flavoring, as the two chunks together will taste bland and repetitive.

Your final pitch to Ms. Halohalo on the SSS is already scheduled for tomorrow, yet you need entirely new designs! Can you satisfy your suppliers' new requirements?

## Input Format

The only line of input contains two space-separated integers $h$ and $w$, the height and width of the SSS.

## Output Format

If there exists no sapin-sapin layout satisfying the conditions, output `Wala` on a single line. Otherwise, output `Mayroon` on a single line.

If there exists a layout, output $h$ lines, each containing $w$ characters. Each of these characters should be one of `V`, `O`, `R`, `W`, `G`, `Y`, `B`. The connected components of same letters must exactly correspond to the shape of the appropriate chunk for that flavor (described in the Explanation section), except possibly for rotation.

If there are multiple layouts, any will be accepted.

## Constraints and Subtasks

| **For all subtasks** |
|---|
| $1 \le h, w \le 888$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **5** | $h = 4, w = 10$ |
| 2 | **5** | $h = w = 8$ |
| 3 | **5** | $h = w = 9$ |
| 4 | **13** | $min(h, w) = 2$ |
| 5 | **14** | $min(h, w) = 5$ |
| 6 | **13** | $min(h, w) = 3$ |
| 7 | **14** | $h, w$ are divisible by 4 |
| 8 | **15** | $h = w$ |
| 9 | **16** | No further constraints. |

## Sample I/O

| Input 1 | Output 1 |
|---|---|
| `3 3` | `Wala` |

| Input 2 | Output 2 |
|---|---|
| `16 16` | `Mayroon`<br>`VVVGGVRRVVVGGVOO`<br>`BVGGVVVRRVGGVVVO`<br>`BBBOOOYYWWWWYYGO`<br>`GWWWWOYYGGBBYYGG`<br>`GGYYBBVGGRBOOOVG`<br>`VGYYBVVVRRBOYYVV`<br>`VVWGBYYBRVVVYYVR`<br>`VRWGGYYBBBVROORR`<br>`RRWBGWWWWORRVORV`<br>`RVWBOOOYYORVVOVV`<br>`VVBBOGGYYOOGVWGV`<br>`GVYYGGRRVVVGGWGG`<br>`GGYYBBBRRVRRGWVG`<br>`OGWWWWBWWWWRRWVV`<br>`OVVVGGVRRVVVGGVO`<br>`OOVGGVVVRRVGGOOO` |

## Explanation

Note that the second sample input is valid for only subtasks 7, 8, and 9.

The shapes of the pieces are as follows:

- Vivid Ube:

```
 V
VVV
```

- Ooh-la-la Orange

```
O
O
OO
```

- Gorgeous Green Grapes

```
 GG
GG
```

- Rad Rambutan

```
RR
 RR
```

- Yummy Langka

```
YY
YY
```

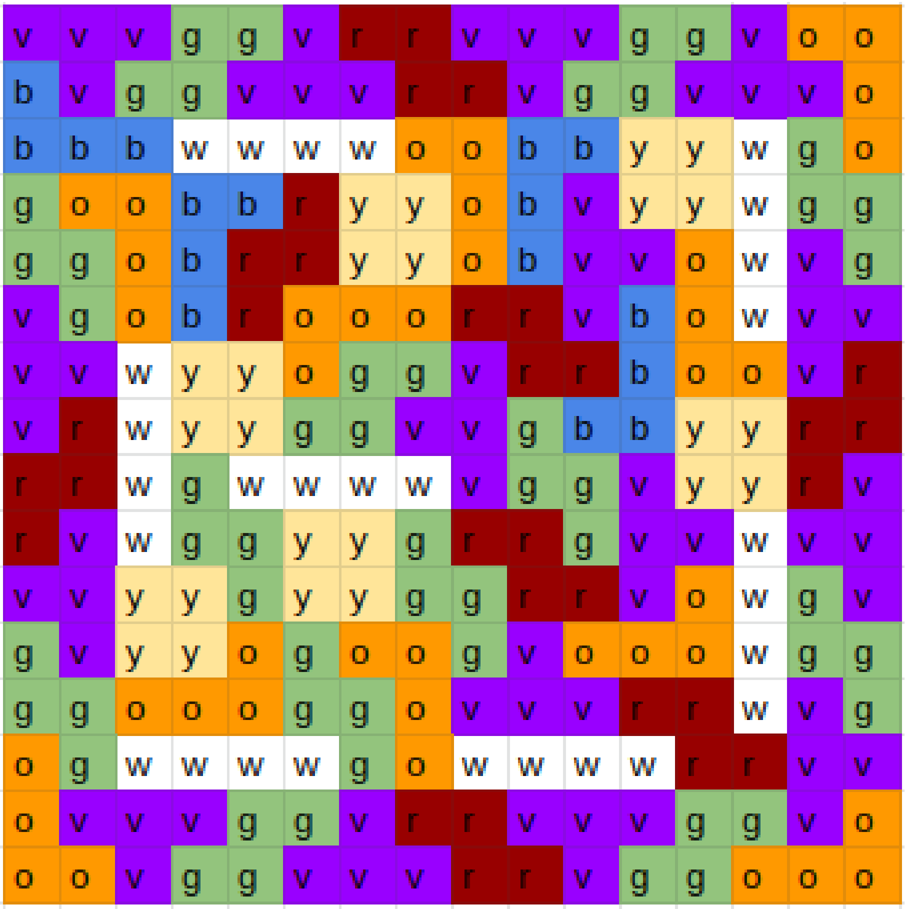
- Blissful Blueberries

```
 B
 B
BB
```

- White Chocolate

```
WWWW
```

Here is an image corresponding to the SSS shown in the second sample input.



| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| v | v | v | g | g | v | r | r | v | v | v | g | g | v | o | o |
| b | v | g | g | v | v | v | r | r | v | g | g | v | v | v | o |
| b | b | b | w | w | w | w | o | o | b | b | y | y | w | g | o |
| g | o | o | b | b | r | y | y | o | b | v | y | y | w | g | g |
| g | g | o | b | r | r | y | y | o | b | v | v | o | w | v | g |
| v | g | o | b | r | o | o | o | r | r | v | b | o | w | v | v |
| v | v | w | y | y | o | g | g | v | r | r | b | o | o | v | r |
| v | r | w | y | y | g | g | v | v | g | b | b | y | y | r | r |
| r | r | w | g | w | w | w | w | v | g | g | v | y | y | r | v |
| r | v | w | g | g | y | y | g | r | r | g | v | v | w | v | v |
| v | v | y | y | g | y | y | g | g | r | r | v | o | w | g | v |
| g | v | y | y | o | g | o | o | g | v | o | o | o | w | g | g |
| g | g | o | o | o | g | g | o | v | v | v | r | r | w | v | g |
| o | g | w | w | w | w | g | o | w | w | w | w | r | r | v | v |
| o | v | v | v | g | g | v | r | r | v | v | v | g | g | v | o |
| o | o | v | g | g | v | v | v | r | r | v | g | g | o | o | o |

# Problem H

## I Love Jose Rizal

Time Limit: 1 second

In lieu of a story, we instead present some real historical facts about our national hero, Jose Rizal.

- Rizal would often have an incredibly tight food budget, due to him blowing his money on other things like books and professional photography sessions.
- Supposedly, during lunch time, Rizal would go out to the restaurants in the food quarter just to *smell* delicious foods. When he returned to the boarding house, the lingering aroma of food on his clothes would have people believe that he really had eaten out!
- Rizal was good friends with another Philippine hero, Antonio Luna. However, the two had almost ended up duelling each other... over a girl! Thankfully, the duel was called off when Luna later sobered up.
- Rizal actually stayed in Japan for quite some time, admiring the Japanese values of industry, courtesy, and cleanliness. He even ended up falling in love with a girl named Seiko Usui, who would be his girlfriend until the time when he had to depart for America.

You are Jose Rizal's biggest fan. Your parents have taken you to the Jose Rizal gift shop!

You have $m$ gift certificates, each worth $a_1, a_2, a_3, \ldots, a_m$ pesos, respectively. You only have one copy of each gift certificate.

The store has $n$ items for sale, each worth $b_1, b_2, b_3, \ldots, b_n$ pesos, respectively. The gift shop only has one stock of each item.

In the future, the Philippines has $d + 1$ denominations of currency for the peso, which are $1, c_1, c_2, c_3, \ldots, c_d$. Note that the 1 peso coin is always a denomination of currency, as it bears the visage of Jose Rizal.

Any number of times, you may cash in **one** gift certificate to buy **one** item, whose price is less than or equal to the value of the gift certificate. You may not buy multiple items with one gift certificate, and neither may you pool together several gift certificates to buy one item.

Also, the gift certificates in this store are special, because the cashier is actually required to give you *change* for your purchase! If you use a gift certificate worth $X$ pesos to buy an item worth $Y$ pesos, with $X \geq Y$, then the cashier must give you $X - Y$ pesos in change.

The cashier has a virtually infinite repository of each denomination of currency. Note that because the 1 peso coin always exists, the cashier will always be able to successfully make change. And that is actually your goal! You don't really want any of the items on sale—you just want to amass as many 1 peso coins as possible! Unfortunately, the cashier is also a self-professed Jose Rizal superfan, and does not want to part with any of their 1 peso coins. So, they will always make change using the **fewest** number of 1 peso coins possible.

For example, suppose the denominations are $\{1, 3, 8\}$ and your change is 10 pesos. Here are all the ways to form 10 pesos with the given denominations:

- $1 + 1 + 8$
- $1 + 3 + 3 + 3$
- $1 + 1 + 1 + 1 + 3 + 3$
- $1 + 1 + 1 + 1 + 1 + 1 + 1 + 3$
- $1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1$

The one that uses the fewest 1 peso coins is $1 + 3 + 3 + 3$, which only uses one 1 peso coin. Note that this uses *more* total coins than $1 + 1 + 8$, but the adversarial cashier prioritizes, above all else, using the fewest number of 1 peso coins.

Naturally, you wonder—what is the maximum possible number of 1 peso coins that you can acquire, assuming that you use your gift certificates optimally? This is a pretty hard problem, so you go to the NOI.PH Discord and ask for help! One of the more experienced contestants explains that this can be reduced to a classic problem known as the assignment problem, which itself can be solved using the well-known Hungarian algorithm.

That's great! The only step left (and the actual meat of this problem) is to fill up the "cost" matrix. Answer the following question for each $1 \leq i \leq m$ and $1 \leq j \leq n$: If you use the $i$th gift certificate to buy the $j$th item, how many 1 peso coins will the adversarial cashier give you as change?

## Input Format

The first line of input contains three space-separated integers $m$, $n$, and $d$—the number of gift certificates, the number of items in the gift shop, and the number of denominations of currency (other than 1).

The second line of input contains $m$ space-separated integers $a_1, a_2, a_3, \ldots, a_m$, the values of your gift certificates.

The third line of input contains $n$ space-separated integers $b_1, b_2, b_3, \ldots, b_n$, the prices of the items in the store.

The fourth line of input contains $d$ space-separated integers $c_1, c_2, c_3, \ldots, c_d$, the denominations of currency (other than 1).

## Output Format

Output a matrix of integers with $m$ rows and $n$ columns. The value in the $i$th row from the top and the $j$th column from the left should be the number of 1 peso coins that the adversarial cashier will give you as change, if you use the $i$th gift certificate to buy the $j$th item. If the $i$th gift certificate *cannot* buy the $j$th item, the corresponding value should be 0.

The values in each row of the matrix should be separated by a single space.

## Constraints and Subtasks

| **For all subtasks** |
|---|
| $1 \leq m, n \leq 400$<br>$1 \leq a_i, b_i \leq 10^9$<br>$1 \leq d \leq 30$<br>$2 \leq c_i \leq 1000$ and the $c_i$ are all distinct. |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **30** | $1 \le a_i, b_i \le 20$ |
| 2 | **15** | $1 \le a_i, b_i \le 2000$ |
| 3 | **20** | $1 \le a_i, b_i \le 10^5$ |
| 4 | **10** | $d = 1$ |
| 5 | **5** | $c = \{301, 296\}$ |
| 6 | **10** | $d = 2$ |
| 7 | **10** | No further constraints. |

## Sample I/O

**Input 1**

```
3 4 2
28 35 10
12 30 18 96
3 8
```

**Output 1**

```
0 0 1 0
0 2 0 0
0 0 0 0
```

**Input 2**

```
3 3 1
100 200 300
40 50 60
1000
```

**Output 2**

```
60 50 40
160 150 140
260 250 240
```

## Explanation

In the first sample input:

- If $i = 1$ and $j = 1$, then $a_i = 28$ and $b_j = 12$. The change is 16 pesos, and the adversarial cashier would give 0 one-peso coins: $16 = 8 + 8$.
- If $i = 1$ and $j = 3$, then $a_i = 28$ and $b_j = 18$. The change is 10 pesos, and the adversarial cashier would give 1 one-peso coin: $10 = 3 + 3 + 3 + 1$.
- If $i = 2$ and $j = 1$, then $a_i = 35$ and $b_j = 12$. The change is 23 pesos, and the adversarial cashier would give 0 one-peso coins: $23 = 8+3+3+3+3+3$.
- If $i = 2$ and $j = 2$, then $a_i = 35$ and $b_j = 30$. The change is 5 pesos, and the adversarial cashier would give 2 one-peso coins: $5 = 3 + 1 + 1$.

- If $i = 2$ and $j = 3$, then $a_i = 35$ and $b_j = 18$. The change is 17 pesos, and the adversarial cashier would give 0 one-peso coins: $23 = 8 + 3 + 3 + 3$.

For all other $i$ and $j$, the value of the gift certificate is too small to afford the corresponding item, so the answer in those cases is 0.

# Problem I

## Nonstop Fitness Training

Time Limit: 3 seconds

The popular influencer NFTbro has become popular for his Nonstop Fitness Training regime. A lot of his followers work in the Crypto Building, a tall *tall* **tall** building with offices on each floor. The floors are labeled $0, 1, 2, 3, \ldots$; the building might as well be infinitely tall!

Two different types of events involving NFTbro take place at certain times of the day:

- `UP x` **Announcement event!**. NFTbro announces that each of his current followers go up $x$ floors in the Crypto Building. Which they all do!
- `FL y` **Follow event!** Someone from the $y$th floor follows NFTbro!

Astrology Girl does not want to follow NFTbro. And she does not want to be around his followers. Unfortunately, she has to stay in Crypto Building until the end of the workday!

After every event, Astrology Girl gets fed up and decides to move to the lowest floor in the building which does not have a single person following NFTbro. You may assume that no NFTbro events take place while Astrology Girl is changing floors.

Astrology Girl starts her day at the lowest floor, Floor 0, of Crypto building. At this point, none of the NFTbro events have happened yet.

Given the chronological order of the NFTbro events, can you tell how many floors Astrology Girl has to travel up or down in order to be in the lowest floor without any NFTbro followers?

## Input Format

The first line of input contains an single integer $n$, the number of events.

Then, $n$ lines follow, each describing an event. Each event description is of the form UP `x` or `FL y`, whose meanings were described in the statement above.

## Output Format

Output $n$ lines. After each event, output the number of floors that Astrology Girl has to travel up or down in order to be in the lowest floor without any NFTbro followers.

## Constraints and Subtasks

| **For all subtasks** |
|---|
| $1 \le n \le 2 \times 10^5$<br>$0 \le x, y \le 10^9$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **15** | $1 \le n \le 50$<br>$0 \le x, y \le 50$ |
| 2 | **20** | $1 \le n \le 6000$<br>$0 \le x, y \le 6000$<br>There will be no `UP` events. |
| 3 | **10** | $1 \le n \le 6000$<br>$0 \le x, y \le 6000$ |
| 4 | **5** | $1 \le n \le 6000$ |
| 5 | **40** | There will be no `UP` events. |
| 6 | **10** | No further constraints. |

## Sample I/O

| Input 1 | Output 1 |
|---|---|
| 6<br>FL 2<br>FL 0<br>FL 1<br>FL 4<br>FL 3<br>FL 1 | 0<br>1<br>2<br>0<br>2<br>0 |

| Input 2 | Output 2 |
|---|---|
| 7<br>FL 2<br>FL 0<br>UP 1<br>FL 0<br>FL 4<br>FL 2<br>UP 2 | 0<br>1<br>1<br>2<br>0<br>3<br>5 |

## Explanation

Recall that Astrology Girl begins on Floor 0. In the first sample input, the first three events go as follows.

- A new follower appears on Floor 2. Astrology Girl does not move.
- A new follower appears on Floor 0. Astrology Girl moves up 1 floor.
- A new follower appears on Floor 1. Astrology Girl moves up 2 floors (since there is also already a follower on Floor 2).
- . . .

In the second sample input, the first three events go as follows.

- A new follower appears on Floor 2. Astrology Girl does not move.
- A new follower appears on Floor 0. Astrology Girl moves up 1 floor.
- All followers move up 1 floor. The followers are now on floors 1 and 3. So, Astrology Girl moves down 1 floor back to floor 0 (which is now follower-free).
- . . .

# Problem J
## Dr Kakaibabe and the Metaverse of Madness

Time Limit: 1 second

Dr

*Dr Kakaiba is the Sysadmin Supreme, meaning he is the warden and maintainer of the frightful and mind-bending reality that is billionaire Smart Zuccini's latest creation—the Metaverse. On the Metaverse, anything that can exist, does exist somewhere—including different versions of Dr Kakaiba in parallel metaverses.*

*He's faced off against evil and edgy Dark! versions of himself, been destroyed by the logic and facts from the massive intellect of Rational! versions of himself, traveled with a TrainConductor! version of himself called Doctor Pasahero (see the problem, "Gagamboy: Walang Daan Pauwi"), and even had a chance to relax and vent his frustrations with an amiable CoffeeShop! version of himself.*

*But by far the most interesting variant is the version he met in the 63rd metaverse—a Female! version of himself, who calls herself Dr. Kakaibabe. As a fellow Sysadmin Supreme, she also keeps her section of the Metaverse safe from threats. The* *two of them have teamed up on many occasions to save the whole Metaverse from collapsing—now is one such time.*

First, you must understand the layout of the Metaverse. The Metaverse consists of $n$ main lobbies, labeled from 1 to $n$. Normal users' digital avatars hang out at these lobbies, and can travel between them using special two-way portals that connect certain pairs of lobbies together. At most one portal connects any two lobbies, and no portal connects a lobby to itself. These portals are also the only way for normal users to travel between different lobbies in the Metaverse.

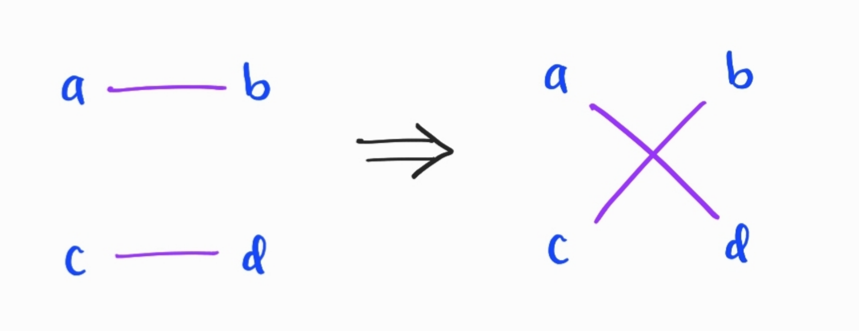
Gagamboy caused a Metaverse-shattering incident which blasted and scrambled all of the portals between lobbies. Once the actual threat that Gagamboy had unleashed had been contained, Dr Kakaiba and Dr Kakaibabe turned to fixing the portals.

As it stands, there are currently $m$ different two-way portals which connect different pairs of lobbies together. Originally, the Metaverse used to be *connected*, meaning any user could start from any lobby and reach any other lobby, just by using the portals. But now, it might not be the case, and some users might have ended up stranded. Dr Kakaiba and Dr Kakaibabe plan on fixing this, but right now the Metaverse is still too fragile, so creating new portals is too risky, as is messing with the number of portals attached to each lobby.

Working together, the duo only have the following operation available to them. Let $(u, v)$ denote a two-way portal that connects lobbies $u$ and $v$.

- Choose four lobbies $a$, $b$, $c$, and $d$ (that are all distinct), such that portals $(a, b)$ and $(c, d)$ currently exist.
- Disconnect the portals $(a, b)$ and $(c, d)$, and reconnect them as the portals $(a, d)$ and $(c, b)$.
- Recall the rules, *"At most one portal connects any two lobbies, and no portal connects a lobby to itself."* If, after this operation, this rule is no longer obeyed, then the Metaverse is destroyed (oh no)

Basically, the operation performs the action shown in the following diagram:



Note that the portals are two-way, so $(a, b)$ is the same as $(b, a)$. However, order matters when declaring the above operation, because it dictates what gets reconnected to what.

The task should be evident, don't you think? Given the portals, determine if it is possible to make the Metaverse connected again, only by using the described operation, and without destroying the Metaverse. Also, if possible, determine the minimum number of operations required, and actually provide such a solution.

## Input Format

The first line of input contains two space-separated integers $n$ and $m$, the number of lobbies and the number of portals.

Then, $m$ lines of input follow, each describing a two-way portal. Each line contains two space-separated integers $a$ and $b$, meaning that a two-way portal connects lobbies $a$ and $b$.

## Output Format

If the task is impossible, output a line containing `NO`. If the task is possible, output a line containing `YES`.

Furthermore, if `YES`, you will be asked to provide such a solution. First, output a line containing a single positive integer $k$, the number of operations performed by your solution.

Then, output $k$ lines, each containing four space-separated integers $a$, $b$, $c$, and $d$, describing the operations that you wish to perform, in order.

Your solution will be considered correct if the Metaverse is connected after all operations have been performed, and if $k$ is the minimum value possible. If there are multiple such optimal solutions, any will be accepted.

## Constraints and Subtasks

| **For all subtasks** |
|---|
| $1 \leq a, b \leq n$<br>For each portal, $a \neq b$, and this is the only portal connecting $a$ and $b$. |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **50** | $1 \leq n, m \leq 50$ |
| 2 | **25** | $1 \leq n, m \leq 3000$ |
| 3 | **25** | $1 \leq n, m \leq 10^5$ |

## Sample I/O

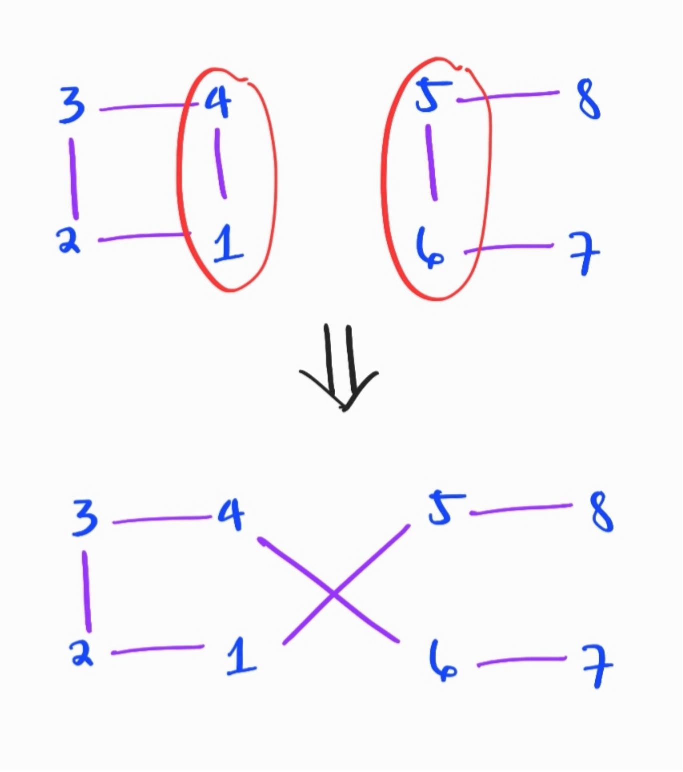
| Input 1 | Output 1 |
|---|---|
| 8 7<br>1 2<br>2 3<br>3 4<br>4 1<br>5 6<br>6 7<br>5 8 | YES<br>1<br>4 1 5 6 |

| Input 2 | Output 2 |
|---|---|
| 2 1<br>1 2 | YES<br>0 |

| Input 3 | Output 3 |
|---|---|
| 4 2<br>1 2<br>3 4 | NO |

## Explanation

The first sample input corresponds to the following scenario, which as you can see, can be resolved in just 1 operation.



In the second sample input, the Metaverse is already connected, so the task is doable in 0 operations.

# Problem K
## Bagsakan

Time Limit: 1 second

*Magbabagsakan dito, in 5, 4, 3, 2...*

As their act for their school's variety show, Alice and Bob and Cindy are going to perform their own remix of the legendary rap song Bagsakan. Aside from the chorus, their remix has $n$ different verses to be sung by the three of them—with stage names "Mirand-A", "Bibo", and "Clock-9".

The verses are labeled $1, 2, 3, \ldots, n$, in sequence. Each verse must be performed by exactly one of Mirand-A or Bibo or Clock-9. Each verse features its own wordplay, rhymes, and of course, sheer speed—all these factors can be summed up in a single numerical *hype value* for each verse.

Each hype value is always a power of 3. Given some sequence of nonnegative integers $e_1, e_2, \cdots, e_n$, you are informed that the hype value of the $i$th verse is exactly equal to $3^{e_i}$.

So that one does not overshadow the others, the trio decides that each of them should get an equal amount of hype, and if not, then they **won't do the performance**.

Formally, let $A$, $B$, and $C$ be the sum of the hype values of the verses given to Mirand-A, Bibo, and Clock-9 respectively. Find any way to distribute the verses between the three of them such that $A = B = C$, or report that this is impossible.

*Magbabagsakan dito, kailangan nang mag-ingat*

*At ang huling bagsakan, ikaw na ang babanat!*

## Input Format

The first line of input contains a single positive integer $n$.

The second line of input contains $n$ space-separated non-negative integers $e_1, e_2, e_3, \ldots, e_n$, encoding the exponents of the hype values of the verses.

## Output Format

First, output a line containing `YES` if the task is possible, and `NO` if it is not.

If `YES`, output another line containing a string of length $n$, encoding a solution that shows how to distribute the verses. For each $i$ from 1 to $n$, let the $i$th character of the string be `A` if Mirand-A gets the $i$th verse (resp. `B` and `C` for Bibo or Clock-9 getting that verse).

If there are multiple solutions, output any of them.

## Constraints and Subtasks

| **For all subtasks** |
|---|
| $1 \leq n \leq 2 \times 10^5$<br>$0 \leq e_i \leq 10^9$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **33** | $1 \leq N \leq 10$ |
| 2 | **10** | $e_i = 0$ for all $i$. |
| 3 | **10** | $0 \leq e_i \leq 1$ for all $i$. |
| 4 | **15** | $0 \leq e_i \leq 25$ for all $i$. |
| 5 | **22** | $0 \leq e_i \leq 2 \times 10^5$ for all $i$. |
| 6 | **10** | No further constraints. |

## Sample I/O

| Input 1 | Output 1 |
|---|---|
| 5<br>0 1 1 0 0 | YES<br>ABCAA |

| Input 2 | Output 2 |
|---|---|
| 2<br>4 5 | NO |

# Problem L
## Gagamboy: Walang Daan Pauwi
Time Limit: 2 seconds

After being bitten by a radioactive spider, Gagamboy has learned to do many of the things a spider can, such as building webs, climbing surfaces, and scaring most humans. However, they have not learned to do the things spiders cannot, such as using public transportation.

For the most part, the area surrounding Gagamboy's house is mostly walkable, so this is not an issue for them. However, when they wish to go to a faraway location, they find themselves stranded. When such scenarios arise, he calls on the help of his magician friend, Doctor Pasahero.

Doctor Pasahero knows $k$ *magic words*, $S_1, S_2, \ldots, S_k$, each of which is a string of lowercase Latin characters. Due to the infinite looping nature of the metroverse, he turns these words into an infinite sequence by setting $S_i = S_{i-k}$ for $i > k$.

Doctor Pasahero can string these magic words together to form spells. However, he must string them together in a very specific manner:

- Spell number 1, $T_1$ is simply $S_1$.
- Spell number 2, $T_2$ is, $T_1$, $S_2$, and $T_1$, in that order.
- Spell number 3, $T_3$, is $T_2$, $S_3$, and $T_2$ in that order.
- $\ldots$

In general, spell number $i$, $T_i$, is $T_{i-1}$, $S_i$, and $T_{i-1}$, in that order.

Using magic he learned from his counterparts elsewhere in the metroverse, Doctor Kakaiba and Doctor Kakaibabe (see "Dr Kakaibabe and the Metaverse of Madness" for more on their adventures), Doctor Pasahero can teleport any subject he chooses to any location he chooses. He must first write the location's name down as a string $Q$ of lowercase Latin characters. Then, after picking a suitable spell $T_x$, he will start reading out the characters of $T_x$ at a rate of one character per millisecond. If, at any point in time, the following conditions are met:

1. Doctor Pasahero said $Q_1$.
2. At some point in time after that, he says $Q_2$.
3. ...

$|Q|$. At some point in time after that, he says $Q_{|Q|}$.

Doctor Pasahero can cast the transportation spell, moving his desired target to his desired location.

Left with no other choice, Gagamboy asks Doctor Pasahero to help him go to each of $w$ locations whose names are $Q_1, Q_2, \ldots, Q_w$. To help Doctor Pasahero prepare the needed rituals, answer the following questions:

- What is the shortest spell he can use to accomplish the goal?
- How long will it take him to transport Gagamboy?

## Input Format

The first line of input contains two space-separated integers: $k$ and $w$.

Each of the next $k$ lines contains a string. The $i$th of these lines contains $S_i$, the $i$th magic word.

Each of the next $w$ lines contains a string. The $i$th of these lines contains $Q_i$, the name of the $i$th location.

## Output Format

For each destination, if Doctor Pasahero cannot take Gagamboy to the destination, output "`We are in the end station now`" in one line.

Otherwise, output two space-separated integers: the minimum $x$ such that Doctor Pasahero can cast the transportation spell with spell $T_x$, and how long it will take him to finish casting it. As this may be an integer too large for mundane computers to compute, output it modulo 14000605.

## Constraints

Let $\sum S$ and $\sum Q$ be the sum of the lengths of all $S_i$ and $Q_i$, respectively.

| For all subtasks |
|---|
| $1 \le k, w \le 10^5$<br>$1 \le \lvert S_i \rvert, \lvert Q_i \rvert \le 10^{18}$<br>$1 \le \sum S, \sum Q \le 3 \cdot 10^5$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **12** | The spell can be casted in at most 20 milliseconds. |
| 2 | **23** | The spell can be casted in at most $10^5$ milliseconds. |
| 3 | **11** | $k = 1$ |
| 4 | **12** | $\lvert Q_i \rvert = 1$ |
| 5 | **18** | $\lvert S_i \rvert = 1$ |
| 6 | **11** | $k \le 5$ |
| 7 | **13** | No further constraints |

## Sample I/O

| Input | Output |
|---|---|
| `5 2`<br>`batanes`<br>`makati`<br>`cebu`<br>`davao`<br>`jolo`<br>`bicol`<br>`malaysia` | `5 96`<br>`We are in the end station now` |

## Explanation

It can be shown that if Doctor Pasahero starts reading out spell number 5:

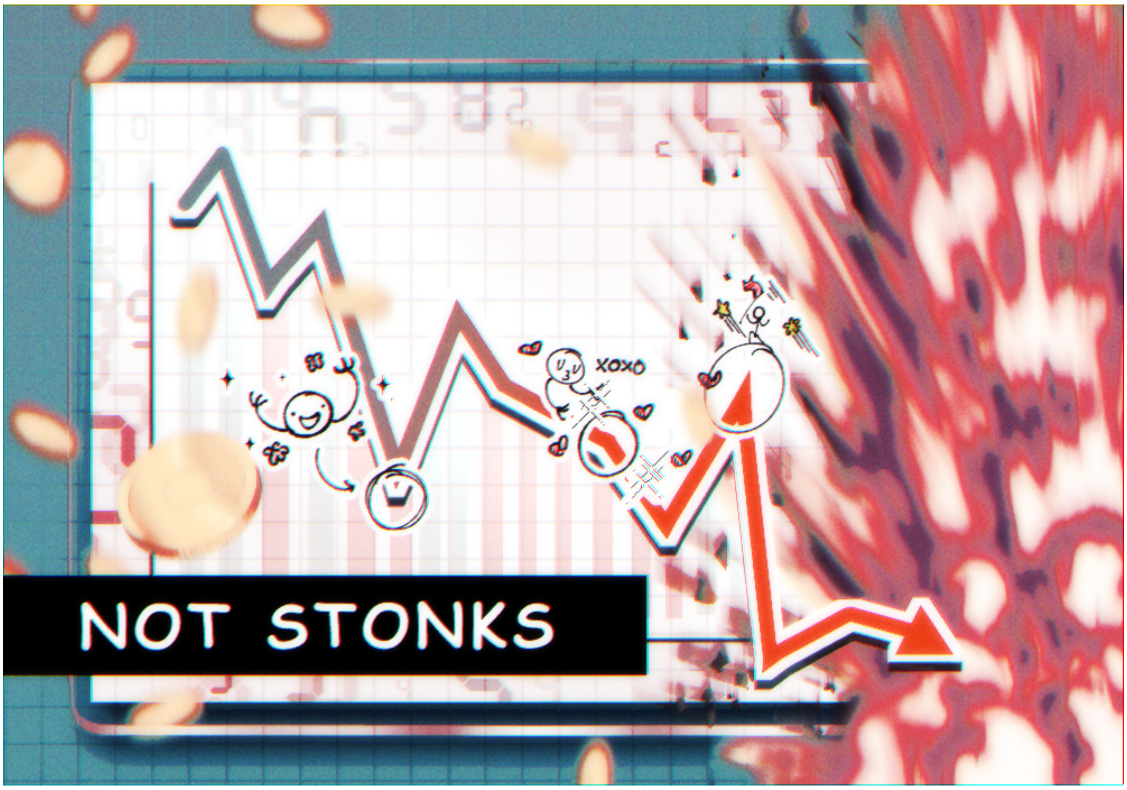
- He says `b` 1ms after starting.
- He says `i` 13ms after starting.
- He says `c` 21ms after starting.
- He says `o` 49ms after starting.
- He says `l` 96ms after starting.

Therefore, he can complete a transportation spell to send Gagamboy to `bicol` after 96ms.

On the other hand, since he will not say `y` under any available spell, he will not be able to send Gagamboy to `malaysia`.

# Problem M
## Easy as Do-Re-Mi

Time Limit: 10 seconds



Psst, hey kid! I see that curious look in your eye, along with that untameable desire to not be left behind on the hottest new trends. Wanna understand what all these fancy NFTs and ETHs and KSPs are? Awesome! *You won't learn any of that here.* None of it's important, anyway! Not if the only thing you care about is getting *filthy stinking rich.*

See, this old song and dance, none of it is new. You don't *actually* need to have skills, or something of value to sell, or anything insightful to say, in order for people to think you're a BIG SHOT. Those are all overrated! Nah, the most important thing is... **presentation**!

With a silver tongue and a bottle full of snake oil, you can get all sorts of suckers to buy the most useless contraptions, so long as you dress it up in sequins and buzzwords. And remember, your best tool for lying is to use statistics!

You see, the quality of math education in this world is absolutely deplorable... which is great news for us! Just smack people with a razzle dazzle of numbers and charts and they'll be too overwhelmed and numerically illiterate to do anything but fold! Most people in civilized society have an innate trust of science and experts.

Abuse that trust!

Here, lemme give you an example. People absolutely love it when *line go up*. Makes them *return to monke*. So they're going to love love love whatever you're selling if you can show that the value of your product is going *up* and *up*. Little do they know... you've been cherry picking the data! But that's just our little secret ;) With this wonderful technique, you can sell people tickets to a ship that's sinking right before their eyes.

And it's not even that hard! Here, let's practice. I'll give you a sequence of $n$ numbers $a_1, a_2, a_3, \ldots, a_n$, representing the value of my newest crypto memecoin over time (People are buying it! What idiots!) Then, I'll give you $q$ different queries to process:

- `? L R` Find any three indices $i, j, k$ such that $L \le i < j < k \le R$ and $a_i < a_j < a_k$. In other words, find any three indices of increasing values within the given range.
- `U i x` Let $a_i$ now have the value of $x$ from now on (at least until I change my mind again). I'm making it all up anyway, so what's the difference?

Even if you don't share my "ethical disposition", you should probably still do this task as a fun theoretical exercise. And, of course, for those sweet sweet points. Extrinsic motivators, am I right?

So, kid, are you up to the challenge?

## Input Format

The first line of input contains two space-separated integers $n$ and $q$.

The second line of input contains $n$ space-separated integers $a_1, a_2, a_3, \ldots, a_n$.

Then, $q$ lines of input follow, each describing a query. Each query is of the form `? L R` or `U i x`, whose meanings were described in the problem statement.

## Output Format

Output one line for every `? L R` query. For each, output any triple of indices $i, j, k$ that fall within $L$ and $R$ (inclusive) such that $a_i < a_j < a_k$. If there are multiple solutions for some query, output any of them. If there are (somehow) no solutions, output `-1 -1 -1` instead.

## Constraints and Subtasks

| For all subtasks |
|---|
| $1 \le n, q \le 2 \times 10^5$<br>$1 \le L \le R \le n$<br>$\lvert R - L \rvert \ge 2$<br>$1 \le i \le n$<br>$1 \le a_i, x \le 10^9$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **20** | $1 \le n, q \le 120$ |
| 2 | **15** | $1 \le n, q \le 2000$ |
| 3 | **22** | $1 \le a_i, x \le 100$ |
| 4 | **20** | $1 \le n, q \le 10^5$<br>There are no `U i x` queries. |
| 5 | **11** | $1 \le n, q \le 10^5$ |
| 6 | **12** | No further constraints. |

## Sample I/O

| Input 1 | Output 1 |
|---|---|
| `7 5`<br>`58 88 94 69 13 25 1`<br>`? 1 4`<br>`? 3 7`<br>`U 3 50`<br>`U 7 99`<br>`? 1 7` | `1 2 3`<br>`-1 -1 -1`<br>`5 6 7` |

## Explanation

For the first query, it can be seen that the values `58, 88, 94`, with indices `1, 2, 3`, form a sequence with increasing values. Hence, the output is `1, 2, 3`.

For the second query, it can be shown that there is no triple in the range with increasing values. Therefore, the output is `-1, -1, -1`. *Tip: You should avoid showing data like this, they don't help sell!*

After performing the two updates, the sequence will now be:

`58, 88, 50, 69, 13, 25, 99`

There are now multiple possible triples in the range `[1, 7]`. One such would be `13, 25, 99`, which has the indices `5, 6, 7`.

*Easy, right?!*

# Problem N
## Kumukutikutitap

Time Limit: 2 seconds

It's Christmas time again: the streets are filled with boys and girls selling lanterns (and cars rushing to make it home for *noche buena*). Jose, and his friends Maria and Charlito, are planning to gather their families somewhere to enjoy the yuletide atmosphere under the glowing lights of the parol festival.

Altogether, Jose has learned that $n$ people will join the gathering. The town they live in can be represented as the section of the Cartesian plane with coordinates with absolute value at most $10^9$ , where the $i$th person lives at $(x_i, y_i)$. To minimize inconvenience for everyone involved, Jose wants to set the meeting to take place at any point with integer coordinates that minimizes the total distance from the meeting point to each person's house. Since Jose lives in Manhattan Street, distance is measured using *Manhattan* distance, where the distance from points $(a, b)$ to $(c, d)$ is

$$|a - c| + |b - d|$$

The planning for the meetup was going well, until suddenly, the city council announced they were also going to have a fireworks display! They have hired $p$ pyrotechnicians, and each assigned them one point with integer coordinates each to launch a dazzling display from! The $i$th pyrotechnician was stationed to the point $(p_i, q_i)$. While Maria would normally appreciate the extra festivity, she wisely points out that the fireworks would pose a safety hazard the all the meetup's attendees, especially the children, can do better without. Therefore, they all agree to avoid setting any of these $p$ locations as their meetup point.

Due to this, it may not be possible to meet up at a point which minimizes total distance. They agree to compromise, and instead, try to meet somewhere that isn't too inconvenient for all of them. Specifically, they will agree on a constant $K$, and will meet somewhere with a total distance not greater than $K$.

Actually, they are still in the process of deciding on what this will be, and have $c$ candidates $k_1, k_2, \ldots, k_c$. For each one, can you inform them of how many places they will have as choices for their meetup site if they choose this candidate as $K$? Note that, due to the bounded size of the town, assuming an infinite value of $K$, they have $S^2 - p$ possible locations, where $S = 2 \cdot 10^{18} + 1$, the number of points on a side of the town.)

## Input Format

The first line of input contains one integer $n$, the number of attendees. The next two lines of input each contain $n$ space-separated integers. The $i$th integer of the firs tline is $x_i$, and the $i$th integer of the second line is $y_i$. The $i$th person lives at $(x_i, y_i)$.

The next line contains one integer $p$, the number of firework launch sites. The next two lines represent the locations of the firework sites, in a similar format as the viewing sites.

The next line contains $c$, the number of candidate values of $K$. The next line contains $c$ space separated integers, representing $k_1, k_2, \ldots, k_c$.

## Output Format

For each candidate $K$, output the total number of possible meetup points.

## Constraints

| **For all subtasks** |
|---|
| $1 \leq n \leq 10^5$<br>$0 \leq p \leq 10^5$<br>$\lvert x_i\rvert, \lvert y_i\rvert, \lvert p_i\rvert, \lvert q_i\rvert \leq 10^9$ for all $1 \leq i \leq n$<br>$1 \leq c \leq 5$<br>$1 \leq k_i \leq 2 \cdot 10^{14}$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **10** | $c = 1$, $k_1$ is the minimum possible total distance. |
| 2 | **15** | $n \leq 50$.<br>If $(a, b)$ is a possible meetup point (ignoring fireworks), then $\lvert a\rvert, \lvert b\rvert \leq 500$. |
| 3 | **12** | $n = 1$ |
| 4 | **12** | $n = 2$ |
| 5 | **16** | $x_i = 0$ |
| 6 | **15** | $\lvert x_i\rvert, \lvert y_i\rvert \leq 200$ |
| 7 | **20** | No further constraints. |

## Sample I/O

| Input | Output |
|---|---|
| <br>5<br>0 2 1 -1 -1<br>0 2 3 1 -1<br>3<br>-1 0 0<br>1 1 2<br>2<br>11 12 | 0<br>2 |

## Explanation

Ignoring the fireworks show, the list of meetup points, sorted by distance, starts:

- Location $(0, 1)$ has total distance 11 from the houses.
- Locations $(-1, 1), (0, 2), (1, 1), (0, 0)$ have total distance 12 from the houses.
- Locations $(1, 2), (1, 0), (-1, 0), (-1, 2)$ have total distance 13 from the houses.

Removing all locations that are used for the fireworks, the resulting list is:

1. Location $(1, 1)$, with total distance 12.
2. Location $(0, 0)$, with total distance 12.
3. Location $(1, 2)$, with total distance 13.

4. ...

(Locations with the same total distance may be listed in arbitrary order. It can be shown that this does not affect the answer.)

# Problem O

## Tinikling 505 - Butsekik's Revenge

Time Limit: 2 seconds

***Butsekik the Butiki is back and ready for Round 2! And he's not playing around anymore!***

Butsekik the Butiki is tired of being unheard, unseen. All that rage has been clawing for years to break out—and it has reached its breaking point. He has been corrupted by the void. His heart has fallen to darkness. He has been consumed by his Shadow, the true self.

Abby and Cody want to save their friend from what he has become. They can communicate with him using Morse Code, but they only remember that -- maps to O, which looks like a 0, and that ... maps to S, which might look like a 5 on a seven-segment display from an old calculator or radio. In another world, they simply would have called out 505 to Butsekik, but right now, such a simple message is not enough! Stronger words are needed to break through to Butsekik and remind him that his friends still care about him.

Abby and Cody call a positive integer a **Generalized Friendship Light Power Heart Love Hope Dreams Life SOS** number if it satisfies the following properties.

- It consists of only the digits 0 and 5.
- If read as a string, it contains $s$ as a substring, where $s$ is a given string that only consists of 0 and 5.

(To eliminate ambiguity with what to do about leading zeroes, the test data will only contain $s$ whose first digit is 5.)

They know that their friend Butsekik is still in there somewhere. They just need to reach out to him! And they're going to need a lot of Generalized Friendship Light Power Heart Love Hope Dreams Life SOS numbers in order to do so.

Given a positive integer $N$, count the number of Generalized Friendship Light

Power Heart Love Hope Dreams Life SOS numbers that are $\leq N$.

Note that $N$ is **huge huge huge** so it will be encoded in the input in a special way. Also, because $N$ is huge, you should only find the answer modulo $p$, where $p$ is a prime number that will be given in the input. You must also answer this question for $T$ different test cases; you are at least assured that the modulus $p$ will be the same across all test cases.

## Input Format

The first line of input contains two space-separated integers $T$ and $p$, the number of test cases, and the prime modulus to be used in all test cases. Then, $T$ lines follow, each describing a test case in the following manner.

First, a line containing the string $s$, the pattern that a number must contain in order to be a Generalized Friendship Light Power Heart Love Hope Dreams Life SOS number. The string $s$ only contains the characters `0` and `5`.

Second, a line containing three space-separated positive integers $a$, $b$, and $c$. We define the $N$ from the problem statement as $N = 10^{a^{b^c}}$.

## Output Format

For each test case, output a line containing a single integer, the count (modulo $p$) of Generalized Friendship Light Power Heart Love Hope Dreams Life SOS numbers that are $\leq N$.

## Constraints and Subtasks

| **For all subtasks** |
|---|
| $1 \leq T \leq 10$<br>$2 \leq p < 10^9$, and $p$ is prime<br>$1 \leq \lvert s \rvert \leq 4$ and its first digit is always `5`<br>$1 \leq a, b, c \leq 10^{18}$ |

| Subtask | Points | Constraints |
|---|---|---|
| 1 | **22** | $1 \le a \le 2$<br>$1 \le b \le 2$<br>$c = 1$ |
| 2 | **28** | $1 \le a \le 2$<br>$1 \le b \le 2$<br>$1 \le c \le 2$ |
| 3 | **21** | $1 \le a \le 10$<br>$1 \le b \le 2$<br>$1 \le c \le 2$ |
| 4 | **11** | $1 \le a \le 10$<br>$1 \le b \le 2$<br>$1 \le c \le 4$ |
| 5 | **10** | $\lvert s \rvert = 1$ |
| 6 | **2** | $p = 998244353$ |
| 7 | **2** | $1 \le b \le 10$<br>$1 \le c \le 18$ |
| 8 | **2** | $\lvert s \rvert \le 3$ |
| 9 | **2** | No further constraints. |

## Sample I/O

| Input 1 | Output 1 |
|---|---|
| `1 998244353`<br>`5`<br>`2 1 1` | `3` |

**Input 2**

```
2 998244353
505
2 2 2
5050
1000000000000000000 1000000000000000000 1000000000000000000
```

**Output 2**

```
55284
281094223
```

## Explanation

For the first sample input, the pattern is simply `5`, with the upper bound being $N = 10^{2^{1^1}} = 100$. The three Generalized Friendship Light Power Heart Love Hope Dreams Life SOS numbers in this range are `5`, `50`, and `55`.