少年圖靈計畫
Young turing Program

0_送分題 (Hello World)

(30分)

時間限制: 1 second
記憶體限制: 256 MB

前言

比賽開始了！

趕快驗證一下，

網路是否設定正確?

上傳競賽程式是否順利?

程式解答是否用 STDOUT 輸出？

都沒問題，30分就到手了！ 繼續 ... 衝！衝！衝！

題目敘述

請寫一個程式輸出Hello World!

輸入格式

本題無需輸入值

輸出格式

$A\backslash\sim Z$[a~z]、空格，以及常用英文符號。

資料範圍

$A\backslash\sim Z$[a~z]、空格，以及驚嘆號 “!”

測試範例

輸入範例 1

(無輸入值)

輸出範例 1

```
Hello World!
```

範例說明

輸入範例1, 無輸入值，簡單而快樂的輸出Hello World!

0_Hello World

(30 points)

Time Limit: 1 second
Memory Limit: 256MB

Introduction

YTP Contest has started!

Let’s verify everything first.

Is the internet setting correct?

Is the source code submission working well?

Do you use STDOUT output for program solutions?

Everything is ready! Go get 30 points now!! Go! Go! Go!

Statement

Please write a program to output Hello World!

Input Format

This problem requires no input.

Output Format

$A\~Z$[a~z], space, and common English punctuation.

Constraints

$A\~Z$[a~z], space, and exclamation mark "!".

Test Cases

Input 1

(no input)

Output 1

```
Hello World!
```

Illustrations

Input 1 has no input, simply output Hello World!

1_藥水調配(Potion Mixing)

(5 分)

時間限制: 1.0 second
記憶體限制: 1024 MB

題目敘述

今天是 YTP 魔法學院的魔藥學課，今天的課堂中藥調製出一款名為「AC 之水」的藥水，傳說中，這款藥水只要喝下，就能在短時間內將許多 Wrong Answer 的程式碼瞬間修改回 Accepted，是個非常強大的藥水。

已知如果要調配出一個完美比例的藥水必須要找出一組完美的 (x, y, z) 使 $a + x = b + y = c + z$。

且 x, y, z 必須是非負整數且滿足 $0 \leq x, y, z \leq n$。

能夠調配出完美比例的配方可能有很多種，請你設計一個程式計算共有幾組 (x, y, z) 能夠調配出完美比例的 AC 藥水。

輸入格式

第一行依序輸入四個正整數 n, a, b, c。

輸出格式

輸出一個整數表示能調配出完美比例的總數。

資料範圍

- $1 \leq n \leq 10^{18}$
- $1 \leq a, b, c \leq 10^{18}$

子任務

- 子任務 1 (5 分): 無額外限制

測試範例

輸入範例 1

```
5 1 2 3
```

輸出範例 1

```
4
```

輸入範例 2

```
38787087876526275 239004788 825710069 773121660
```

輸出範例 2

```
38787087289820995
```

範例說明

在範例一中：

(2, 1, 0)
(3, 2, 1)
(4, 3, 2)
(5, 4, 3)

共四組答案，因此輸出 4。

1_Potion Mixing

(5 points)

Time Limit: 1.0 second
Memory Limit: 1024 MB

Statement

Today is the Potions class at YTP Magic Academy. In today's class, we need to brew a potion called "AC Water". Legend has it that this potion, when consumed, can instantly transform many Wrong Answer codes back to Accepted in a short time, making it a very powerful potion.

It is known that to brew a potion with perfect proportions, we must find a perfect set of (x, y, z) such that $a + x = b + y = c + z$.

Moreover, x, y, z must be integers satisfying $0 \leq x, y, z \leq n$.

There may be many different recipes that can produce perfect proportions. Please design a program to calculate how many sets of (x, y, z) can brew the perfect AC potion.

Input Format

The first line contains four positive integers n, a, b, c in that order.

Output Format

Output an integer representing the total number of perfect ratio combinations that can be blended.

Constraints

- $1 \leq n \leq 10^{18}$
- $1 \leq a, b, c \leq 10^{18}$

Subtasks

- Subtask 1 (5 points): No additional constraints

Samples

Sample Input 1

```
5 1 2 3
```

Sample Output 1

```
4
```

Sample Input 2

```
38787087876526275 239004788 825710069 773121660
```

Sample Output 2

```
38787087289820995
```

Illustrations

In Sample 1：

(2, 1, 0)
(3, 2, 1)
(4, 3, 2)
(5, 4, 3)

Output 4

2_易榮數(Honor-guise Number)

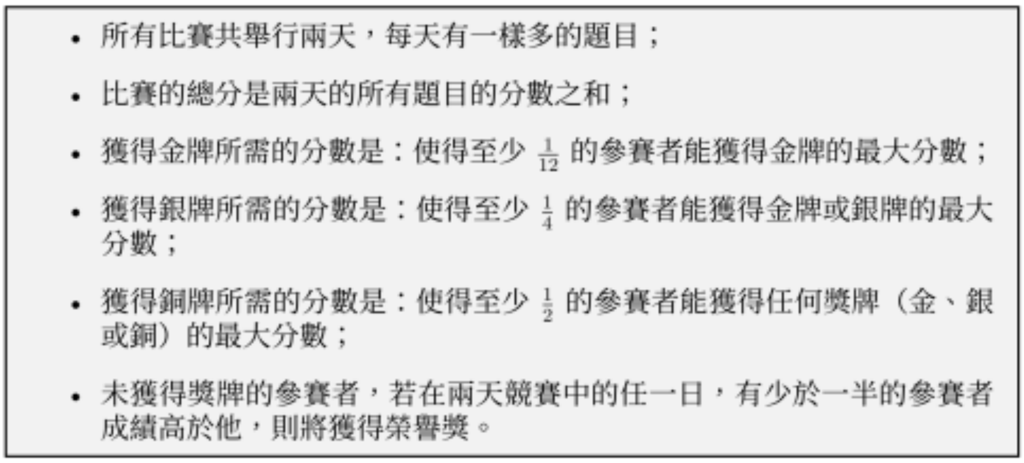
(5 分)

時間限制: 1.0 second
記憶體限制: 1024 MB

題目敘述

你聽過「易容術」嗎？他是一個藉由化妝等方式將一個人的臉型、身形等完全改變成另一種樣子的技術。從小讀武俠小說的小 Y 對這個名詞自然並不陌生，甚至已經對其深深癡迷了。不過，作為一個算法競賽選手，他雖然很有興趣，但在現實中沒什麼機會學到與使用。
因此他把焦點轉移到了「易榮數」上。至於「易榮數」呢，顧名思義，就是最容易獲得榮譽獎的數。也就是說，一場比賽的所有榮譽獎獲得者裡面，最常見的總分就會被小 Y 定義成這場比賽的「易榮數」。

在小 Y 的國家，他們對於算法競賽的舉辦與授獎有以下規定：



- 所有比賽共舉行兩天，每天有一樣多的題目；
- 比賽的總分是兩天的所有題目的分數之和；
- 獲得金牌所需的分數是：使得至少 $\frac{1}{12}$ 的參賽者能獲得金牌的最大分數；
- 獲得銀牌所需的分數是：使得至少 $\frac{1}{4}$ 的參賽者能獲得金牌或銀牌的最大分數；
- 獲得銅牌所需的分數是：使得至少 $\frac{1}{2}$ 的參賽者能獲得任何獎牌（金、銀或銅）的最大分數；
- 未獲得獎牌的參賽者，若在兩天競賽中的任一日，有少於一半的參賽者成績高於他，則將獲得榮譽獎。

對，你會發現榮譽獎好像是一個蠻爛的獎項，而這確實，但是小 Y 才不管呢。

因此小 Y 收集了很多場比賽的計分板，準備一旦算出「易榮數」，他就要在下一次比賽中努力拿到剛好這個分數。但是他太懶了，以致於他不想一個一個算那些比賽的「易榮數」，因此他希望你能寫一支程式來算出來並把結果告訴他。

對了，如果有很多個總分在榮譽獎得主裡面一樣常見，那麼他想要的是總分最小的，因為這樣子的分數看起來最容易拿到。而如果沒有人可以拿到榮譽獎則輸出一行 -1 即可。

輸入格式

輸入的第一行是兩個用空白隔開的正整數 N, M，代表這個比賽的參賽者有 N 個人，並且一天總共有 M 題。

接下來 N 行，每行有 $2M$ 個用空白隔開的正整數 $A_{i,1}, A_{i,2}, \ldots, A_{i,2M}$，代表其中一個人的成績，前 M 個正整數代表他的第一天的每題的分數，後 M 個正整數代表他的第二天的每題的分數。

輸出格式

輸出一個正整數代表能夠所有拿到榮譽獎的人中最常見的總分（易榮數），如果有多個榮譽獎的總分一樣常見，輸出總分最小的。如果沒有人拿到榮譽獎，輸出 -1。

資料範圍

- $2 \leq N \leq 2 \times 10^5$
- $1 \leq M \leq 10^5$
- $2 \leq N \cdot M \leq 2 \times 10^5$
- $1 \leq A_{i,j} \leq 10^9$

子任務

- 子任務 1 (5 分): 無額外限制

測試範例

輸入範例 1

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

輸出範例 1

```
29
```

輸入範例 2

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

輸出範例 2

```
-1
```

範例說明

在第一筆範例測試資料中，四個人的總分分別是 $31, 29, 37, 32$ 分，因此後兩者會拿到金/銀牌，而由於第一個人的第二天的總成績是 18 分，第二個人第一天的總成績是 14 分，都剛好贏了兩個人，因此兩個人都能拿到榮譽獎。也因此 29 跟 31 這兩個分數出現相同次數，此時要輸出比較小的，故輸出 29。

在第二筆範例測試資料中，沒有人能拿到榮譽獎，因此輸出 -1。

2_Honor-guise Number

(5 points)

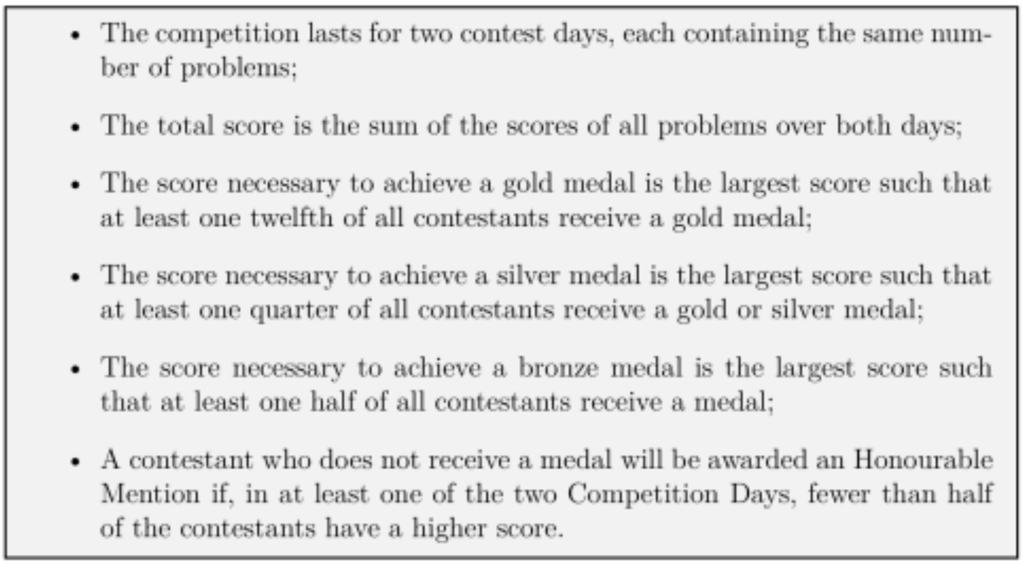
Time Limit: 1.0 second
Memory Limit: 1024 MB

Statement

Have you heard of "disguise arts"? It’s a technique that allows someone to completely change their appearance -- such as face shape or body build -- through makeup and other means. Yuuki, who grew up reading martial arts novels, is naturally fascinated by the idea. But as an competitive programmer, they don’t really have the opportunity to learn or use such a skill in real life.

So, they shifted their focus to something else: the Honor-guise Number. As the name suggests, it's the number that makes it easiest to receive an Honorable Mention. Specifically, it is defined as **the most frequent total score** among all contestants who received an Honorable Mention in a competition.

In Yuuki’s country, the rules for awarding medals and Honorable Mentions in competitive programming contests are as follows:



- The competition lasts for two contest days, each containing the same number of problems;
- The total score is the sum of the scores of all problems over both days;
- The score necessary to achieve a gold medal is the largest score such that at least one twelfth of all contestants receive a gold medal;
- The score necessary to achieve a silver medal is the largest score such that at least one quarter of all contestants receive a gold or silver medal;
- The score necessary to achieve a bronze medal is the largest score such that at least one half of all contestants receive a medal;
- A contestant who does not receive a medal will be awarded an Honourable Mention if, in at least one of the two Competition Days, fewer than half of the contestants have a higher score.

Yes, it does seem like Honorable Mentions are rather lame awards -- and that’s true -- but Yuuki doesn’t care.

They have collected the scoreboards of many past contests, and once they find the Honor-guise Number for each one, they plan to aim for exactly that score in the next competition. However, they are too lazy to manually compute them all. So, they hope you can write a program that finds the Honor-guise Number for each contest and reports it to them.

By the way, if there are multiple total scores that are equally common among Honorable Mentions, Yuuki wants the **smallest** one — since it looks the easiest to achieve. Also, if no contestant qualifies for an Honorable Mention, output a single line containing -1.

Input Format

The first line contains two positive integers N and M, representing the number of contestants and the number of problems per day in the contest.

Each of the next N lines contains $2M$ positive integers, denoted as $A_{i,1}, A_{i,2}, \ldots, A_{i,2M}$, representing the scores of a single contestant. The first M integers represent the scores for each problem on Day 1, and the last M integers represent the scores for Day 2.

Output Format

Print a single integer — the Honor-guise Number, i.e., the most frequent total score among all contestants who received an Honorable Mention.

If multiple scores appear the same number of times, print the smallest such score.

If no contestant received an Honorable Mention, print -1.

Constraints

- $2 \leq N \leq 2 \times 10^5$
- $1 \leq M \leq 10^5$
- $2 \leq N \cdot M \leq 2 \times 10^5$
- $1 \leq A_{i,j} \leq 10^9$

Subtasks

- Subtask 1 (5 points): No additional constraints

Samples

Sample Input 1

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

Sample Output 1

```
29
```

Sample Input 2

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

Sample Output 2

```
-1
```

Illustrations

In the first sample test case, the total scores of the four contestants are 31, 29, 37, and 32, respectively. Therefore, the last two contestants receive the gold and silver medals. The first contestant's total score on the second day is 18, and the second contestant's total score on the first day is 14, so both of which are higher than the scores of exactly two other contestants on that day. As a result, both of them receive Honorable Mentions. Among the contestants who received Honorable Mentions, the scores 29 and 31 each appear once. Since both scores appear the same number of times, the smaller one, 29, is output.

In the second sample test case, no contestant qualifies for an Honorable Mention, so the output is -1.

3_PCC 和 Tetris

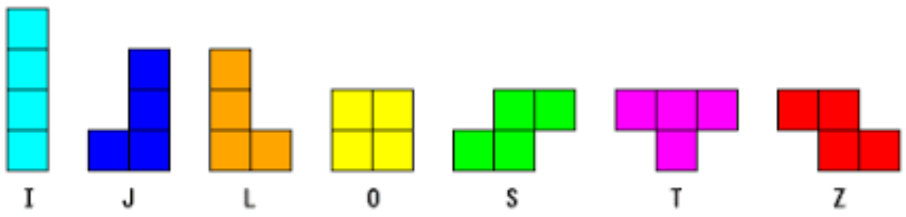
(10 分)

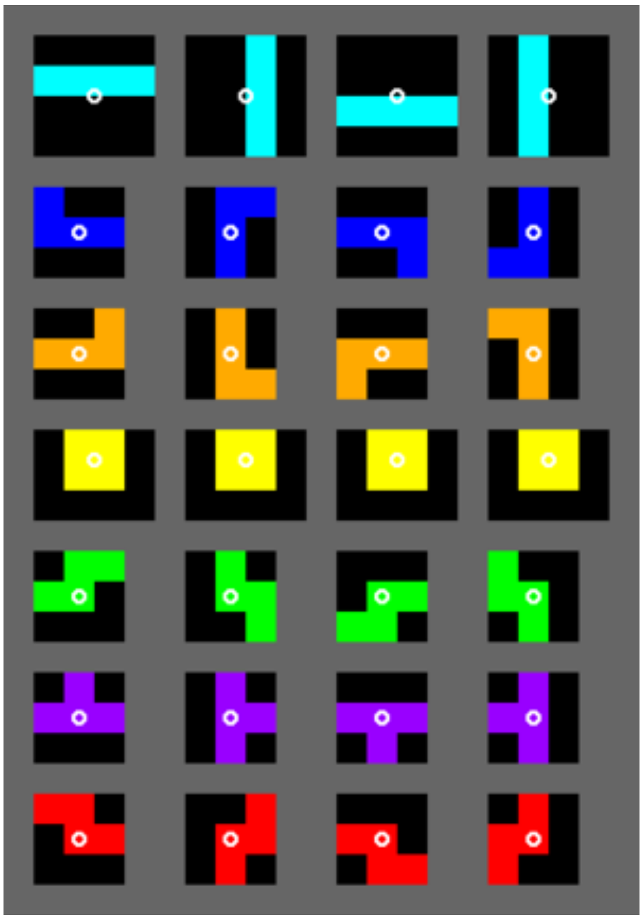
時間限制: 1.0 second
記憶體限制: 1024 MB

題目敘述

在台灣某個大學中，有很多人喜歡打俄羅斯方塊遊戲 (Tetris)，於是 PCC 決定要用俄羅斯方塊 (Tetris) 的積木設計一個遊戲，讓大家挑戰看看。

Tetris 的積木大小都是四格，形狀有 7 種，並且可以進行旋轉，但是不能翻轉:





PCC 製作了一個大小為 $2 \times n$ 的木盒子，還有許多個 Tetris 積木，你的目標是要把 Tetris 積木全部都放到盒子裡。每一關 PCC 會給定每種積木需要用幾個，壞心眼的 PCC 有時候也會給你無法完成的關卡來欺騙你，如果可以的話你需要告訴 PCC 是怎麼擺放的，不行的話需要告訴 PCC 無法完成。

輸入格式

第一行輸入一個正整數 N，代表盒子的長度。

第二行會輸入七個正整數，代表積木 I, J, L, O, S, T, Z 的數量。

由於 PCC 很奇怪，所以每個積木的數量都是奇數個。

輸出格式

如果無法擺放所有積木到盒子裡，輸出一行 "`No`" (不含引號)

如果可以擺放所有積木到盒子裡，輸出一行 "`Yes`" (不含引號)

接著輸出兩行，代表你怎麼把積木放進盒子的，擺放的方法如下:

- 使用整數為每個 tetris 方塊編號，同一個方塊佔據的所有格子必須使用相同的編號。
- 你可以自由選擇編號，只要從 1 開始，並依序遞增即可。
- 每一行的數字以空格隔開
- 可以參考範例測資的格式

資料範圍

- $1 \leq N \leq 10^5$
- $I + J + L + O + S + T + Z \leq 5 \times 10^4$

子任務

- 子任務 1 (10 分): 無額外限制

測試範例

輸入範例 1

```
21
1 1 1 1 1 1 1
```

輸出範例 1

```
Yes
0 1 1 2 2 2 0 0 0 0 4 4 4 5 5 0 6 6 0 7 0
1 1 0 2 0 0 3 3 3 3 0 0 4 0 5 5 6 6 7 7 7
```

輸入範例 2

```
4
1 1 1 1 1 1 1
```

輸出範例 2

```
No
```

範例說明

第一個範例測資中，可以將全部的積木都放進盒子裡，因此輸出 "`Yes`"，並且把積木依序放入。

第二個範例測資中，無法將積木全部放入，因此輸出 "`No`"。

3_PCC and Tetris

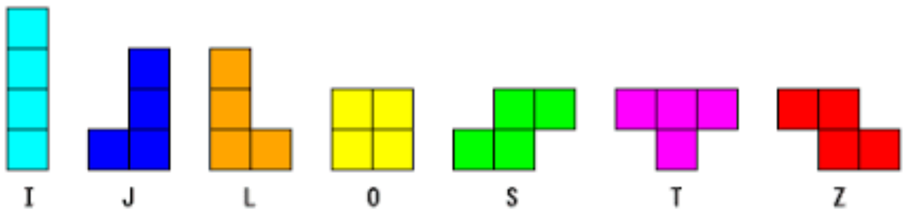
(10 points)

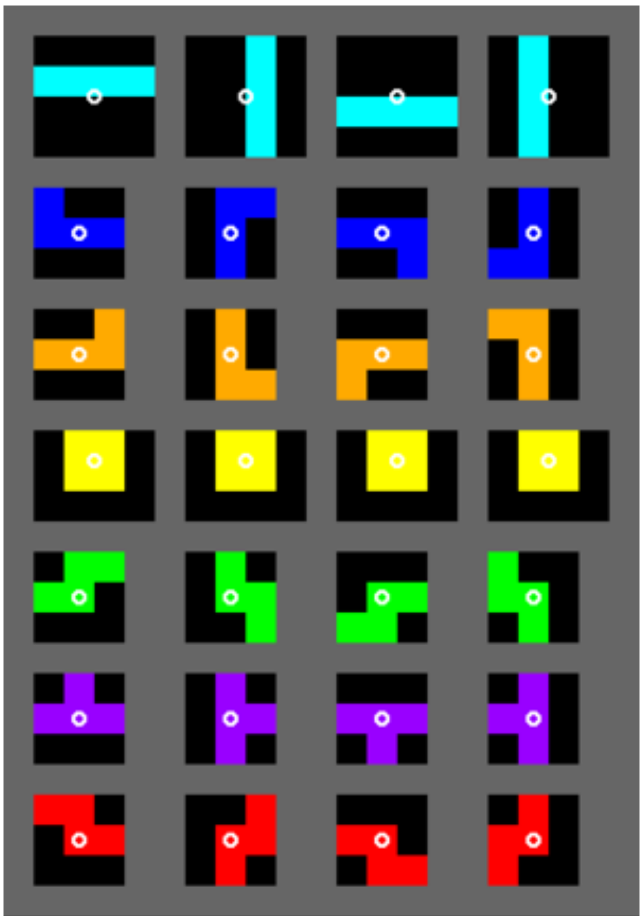
Time Limit: 1.0 second
Memory Limit: 1024 MB

Statement

In a certain university in Taiwan, lots of students are into Tetris, so PCC came up with a challenge: use Tetris blocks to design a new puzzle game and see who can beat it.

All Tetris pieces are made of four squares, and there are seven classic shapes, they can be rotated, but not flipped:





PCC has built a wooden box that's $2 \times n$ in size, and they've got an unlimited supply of Tetris blocks. Your job is to pack all the given blocks into the box. For each level, PCC will tell you exactly how many of each piece you must use. Sometimes, PCC will hand you a configuration that can't possibly fit. If a level is solvable, you need to explain how you'd place the pieces; if it's impossible, you should tell PCC that there's no way to fit them all.

Input Format

The first line contains a positive integer N, denoting the length of the box.

The second line contains seven positive integers, giving the counts of the I, J, L, O, S, T, and Z tetrominoes.

Because PCC is odd, each of those counts will always be an odd number.

Output Format

If it is impossible to fit all the tetrominoes into the box, print "`No`" on the first line

If it is possible to fit them all, print "`Yes`" on the first line, and then two more lines describing your placement:

- Use integers to label each tetromino; all cells occupied by the same piece must have the same label.
- You may choose the labels arbitrarily as long as they start from 1 and increase consecutively.
- Each number on a line is separated by a space.
- You may refer to the sample test case for the exact format.

Constraints

- $1 \leq N \leq 10^5$
- $I + J + L + O + S + T + Z \leq 5 \times 10^4$

Subtasks

- Subtask 1 (10 points): No additional constraints

Samples

Sample Input 1

```
21
1 1 1 1 1 1 1
```

Sample Output 1

```
Yes
0 1 1 2 2 2 0 0 0 0 4 4 4 5 5 0 6 6 0 7 0
1 1 0 2 0 0 3 3 3 3 0 0 4 0 5 5 6 6 7 7 7
```

Sample Input 2

```
4
1 1 1 1 1 1 1
```

Sample Output 2

```
No
```

Illustrations

In the first sample test case, all of the pieces can be placed into the box, so output "`Yes`" and place pieces in order.

In the second sample test case, it is not possible to fit all of the pieces into the box, so output "`No`".

4_停車場(Parking Lot)

(10 分)

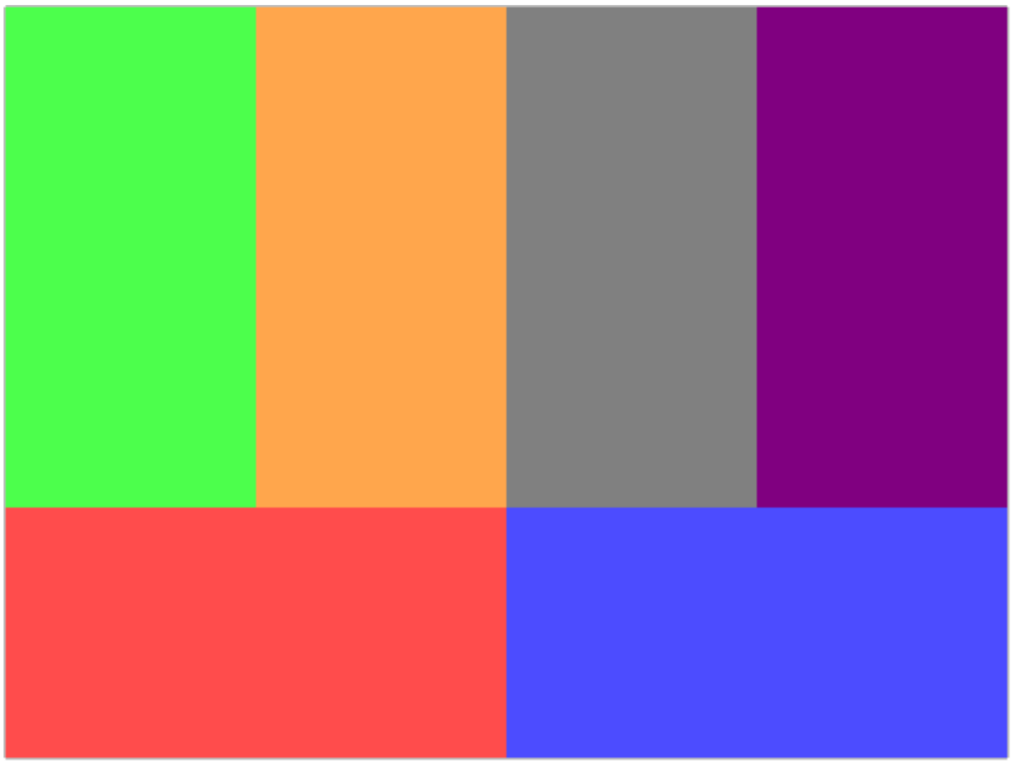
時間限制: 1.0 second
記憶體限制: 1024 MB

題目敘述

NasaLee 在打敗 Vegetable （如果不知道發生什麼事，請看去年的題目）並且扭轉政治局勢後，首先推出了 CSIE-Bike (Cycle for Subversion, Independence, and Emancipation，即「反抗、獨立、解放腳踏車」) 服務來鞏固他的商業地位。而他第一個要處理的問題，就是腳踏車的停放。

這個問題其實並不困難，因為 NasaLee 最近得到了一個停車場，作為政變成功的禮物，沒錯，禮物除了是一個整數序列之外還可以是一個停車場。停車場的長度是 N，寬度是 M，裡面每一個格子都可以被劃分作為停車位。一個腳踏車會佔據**相鄰**的兩格。具有商業頭腦的 NasaLee 不希望停車場有任何位置被浪費，因此他希望**所有格子**都屬於一個停車位。出於某種宗教原因，他相信兩個**同向**且**短邊相接**的停車位會招來厄運，因此他想要避免這件事情發生。

以下是一個 $N = 3, M = 4$ 的規劃範例。



如上圖所示，不同顏色代表不同停車位：

- 紅色與藍色的停車位是**同向且短邊相接**，這是禁止的配置。注意到停車位並沒有面向左邊或是面向右邊的區別。
- 綠色與紅色的停車位**並非同向**，因此也屬於合法配置。
- 紫色與灰色的停車位雖然同向但是**沒有短邊相接**，所以也是合法的。

因為上圖的配置的紅色跟藍色停車位是禁止配置，因此整個配置是不合法的。

現在，由於 NasaLee 的軍國要事過於繁忙，他希望你能幫忙規劃出一種不會招來厄運又把所有格子都使用到的停車位劃分方式。

輸入格式

輸入只有一行，包含以空白分隔的兩個整數 N, M，代表停車場的長寬。

輸出格式

如果不存在一個方式滿足需求，那麼在第一行輸出 `"NO"`（不含引號）。
否則在第一行輸出 `"YES"`（不含引號），接下來輸出 N 行，每行包含 M 個以空白分隔的正整數，代表你幫他規劃的停車場規劃，其中屬於同一個停車位的格子用相同的數字，不同車位的格子用不同的數字表示。

這些格子的編號必須是從 1 開始的連續正整數，否則 NasaLee 會覺得你想敲詐他而不高興。

資料範圍

- $1 \leq N, M \leq 2000$

子任務

- 子任務 1 (10 分): 無額外限制

測試範例

輸入範例 1

```
2 3
```

輸出範例 1

```
YES
1 2 3
1 2 3
```

輸入範例 2

```
3 3
```

輸出範例 2

```
NO
```

輸入範例 3

```
4 4
```

輸出範例 3

```
YES
1 2 5 5
1 2 6 6
3 3 7 8
4 4 7 8
```

輸入範例 4

```
5 4
```

輸出範例 4

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

4_Parking Lot

(10 points)

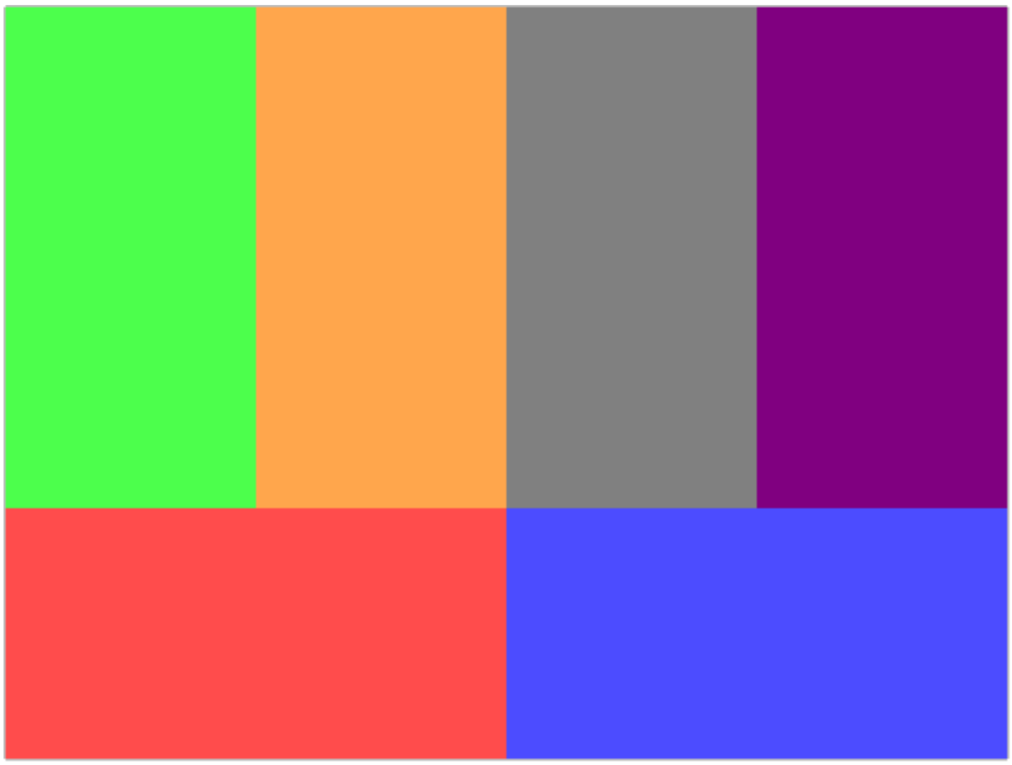
Time Limit: 1.0 second
Memory Limit: 1024 MB

Statement

After defeating Vegetable (if you don't know what happened, please refer to last year's problem) and overturning the political regime, NasaLee launched the CSIE-Bike service (Cycle for Subversion, Independence, and Emancipation) to strengthen his commercial position. The first issue he needs to deal with is bicycle parking.

This problem is actually not very difficult, because NasaLee recently received a parking lot as a gift for his successful coup. That's right — a gift can be more than just an integer sequence; it can even be a parking lot. The parking lot has length N and width M, and each cell in the lot can be designated as a parking space. Each bicycle occupies two **adjacent** cells. With his business mindset, NasaLee doesn't want any space in the parking lot to be wasted, so he wants **every cell** in the lot to be part of a parking space. Due to certain religious beliefs, he thinks that two parking spaces that are **aligned and touch along their short side** will bring misfortune, so he wants to avoid this situation.

Below is an example layout with $N = 3$, $M = 4$:



As shown in the figure, different colors represent the areas belonging to different parking spaces:

- The red and blue parking spaces are **aligned and touching along their short side**, which is a forbidden configuration. Note that parking spaces don't distinguish between facing left or right -- they are still considered parallel.
- The green and red spaces are **not aligned**, so this is a valid configuration.

- The purple and gray spaces are aligned but **do not touch along their short side**, so this is also valid.

Since the red and blue parking spaces in the figure form a forbidden configuration, the entire layout is considered invalid.

Now, because NasaLee is too busy with military affairs, he hopes you can help him design a way to divide the parking lot into spaces such that no misfortune will occur and every cell is used.

Input Format

The input contains a single line with two space-separated integers N and M, representing the length and width of the parking lot.

Output Format

If it is impossible to fill the parking lot without violating the rules, output `"NO"` (without quotes) on the first line.

Otherwise, output `"YES"` on the first line, followed by N lines. Each of these lines should contain M space-separated positive integers. The grid represents your parking plan, where each number identifies a parking space:
Cells with the same number belong to the same parking space (i.e., the same bicycle). Different numbers represent different parking spaces.
These cells should be labeled with consecutive positive integers starting from 1, or NasaLee will think you're trying to extort him and get upset.

Constraints

- $1 \leq N, M \leq 2000$

Subtasks

- Subtask 1 (10 points): No additional constraints

Samples

Sample Input 1

```
2 3
```

Sample Output 1

```
YES
1 2 3
1 2 3
```

Sample Input 2

```
3 3
```

Sample Output 2

```
NO
```

Sample Input 3

```
4 4
```

Sample Output 3

```
YES
1 2 5 5
1 2 6 6
3 3 7 8
4 4 7 8
```

Sample Input 4

```
5 4
```

Sample Output 4

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

5_Two Elevators

(3 分 / 4.5 分 / 7.5 分)

時間限制: 1.0 second
記憶體限制: 512 MB

題目敘述

小傑和奇犽來到了「天空鬥技場」，一座擁有 10^9 層樓的高塔，每一層都會有一位樓主等著其他人來挑戰。以他們的實力，可以輕鬆打敗所有樓主。但是，想要快速晉級並賺取豐厚獎金，就必須高效利用競技場內僅有的**兩部專用電梯**。這些電梯由念能力驅動，每一次移動都會消耗使用者的**念能力**。

小傑和奇犽從他們的師傅手中獲得了一份天空鬥技場的挑戰順序，他們一共要挑戰其中的 N 個樓層，分別是 $F_1, F_2, \ldots, F_N$，每層樓最多挑戰一次。他們必須照這個順序到達各個樓層進行挑戰，挑戰 F_{i+1} 層之前必須先挑戰完 F_i 層。因為他們本身的實力強悍，只需要小傑和奇犽**其中一位**到達要挑戰的樓層即可完成挑戰。所以如何分配各個樓主的挑戰人選，來降低移動時的念能力的消耗就變得極其重要，電梯的念力消耗法則如下：

一台電梯從 x 樓層移動到 y 樓層 $(x \neq y)$，需要消耗 $f(x, y)$ 數量的念能力，遵循以下法則：

$$f(x, y) = A + B \cdot \max(0, y - x) + C \cdot \max(0, x - y) + D \mid y - x \mid$$

- A：每次電梯被指令移動，都需要消耗的基礎念力。
- $B \cdot \max(0, y - x)$：當電梯向上移動時，每層樓會額外消耗的念力。
- $C \cdot \max(0, x - y)$：當電梯向下移動時，每層樓會額外消耗的念力。
- $D \mid y - x \mid$：無論電梯向上或向下，每層樓都會累積消耗的念力。

小傑和奇犽以及兩部電梯最初都在天空鬥技場的 **1 樓**。小傑和奇犽必須精打細算，規劃出最佳的電梯移動路線，依序完成所有樓主的挑戰。

請你幫助小傑和奇犽，找出兩個人**最少的念力消耗總和**是多少。

輸入格式

輸入有兩行：

第一行包含五個整數 N, A, B, C, D，分別代表要經過的樓層數、以及能量計算公式中的常數。

第二行包含 N 個整數 $F_1, F_2, \ldots, F_N$，代表依序要經過的指定樓層，保證數字不會重複。

輸出格式

輸出一個整數，代表挑戰完 N 個樓層所需的最少念能力。

資料範圍

- $1 \leq N \leq 3000$
- $0 \leq A, B, C, D \leq 5000$
- $1 \leq F_i \leq 10^9$，保證 F_i 兩兩相異。

- 所有輸入數字皆為整數。

子任務

- 子任務 1 (3 分): $1 \le N \le 18$
- 子任務 2 (4.5 分): $1 \le N \le 500, 1 \le F_i \le 500$
- 子任務 3 (7.5 分): 無額外限制

測試範例

輸入範例 1

```
4 0 0 0 1
1 10 2 9
```

輸出範例 1

```
11
```

輸入範例 2

```
5 156 125 493 201
101818827 550809474 17607798 702981427 349052006
```

輸出範例 2

```
342962899286
```

範例說明

- 一開始兩台電梯都停在**1樓**
- 第一筆測資中，最佳的行動方法如下：

  $F_1 = 1$，兩台電梯都在1樓，不用做任何移動就可以挑戰完畢，花費：0

  $F_2 = 10$，移動其中一台電梯到10樓，花費： $|1 - 10| = 9$

  $F_3 = 2$，移動原本在1樓的電梯，花費： $|1 - 2| = 1$

  $F_4 = 9$，移動原本在10樓的電梯，花費： $|10 - 9| = 1$

  總共花費：$0 + 9 + 1 + 1 = 11$

5_Two Elevators

(3 points / 4.5 points / 7.5 points)

Time Limit: 1.0 second
Memory Limit: 512 MB

Statement

Gon and Killua arrive at **Heavens Arena arc**, a towering structure with 10^9 floors. On each floor awaits a challenger. With their incredible strength, they can easily defeat any opponent. However, to advance quickly and earn generous rewards, they must make efficient use of the arena's only **two dedicated elevators**. These elevators are powered by Nen, and each movement consumes a certain amount of **Nen energy**.

Gon and Killua have decided on a specific **sequence** of distinct floors they must visit: $F_1, F_2, \ldots, F_N$. They must challenge the floor masters in this order — that is, they cannot proceed to challenge F_{i+1} until F_i is completed. Since either Gon **or** Killua reaching a target floor is sufficient, planning how the elevators are used becomes crucial. The **Nen energy** consumed when an elevator moves from floor x to floor y $(x \neq y)$ is calculated by the following function:

$$f(x, y) = A + B \cdot \max(0, y - x) + C \cdot \max(0, x - y) + D \cdot |y - x| \tag{1}$$

Where:

- A: Base Nen energy cost each time an elevator is moved.
- $B \cdot \max(0, y - x)$: Extra energy cost per floor when moving **up**.
- $C \cdot \max(0, x - y)$: Extra energy cost per floor when moving **down**.
- $D \cdot |y - x|$: General energy cost per floor moved, regardless of direction (for maintaining structural stability and resisting air currents).

Gon, Killua, and both elevators **start on floor 1**. Your task is to help them plan the most energy-efficient use of the elevators to complete all N challenges in order.

Input Format

The input consists of two lines:

The first line contains five integers N, A, B, C, D, representing the number of floors to visit and the constants in the energy consumption formula, respectively.

The second line contains N distinct integers $F_1, F_2, \ldots, F_N$, representing the target floors to visit in sequential order.

Output Format

Output a single integer representing the minimum total Nen consumed to complete the task.

Constraints

- $1 \leq N \leq 3000$

- $0 \le A, B, C, D \le 5000$
- $1 \le F_i \le 10^9$
- All input numbers are integers.
- It is guaranteed that F_i are pairwise distinct.

Subtasks

- Subtask 1 (3 points): $1 \le N \le 18$
- Subtask 2 (4.5 points): $1 \le N \le 500$, $1 \le F_i \le 500$
- Subtask 3 (7.5 points): No additional constraints

Samples

Sample Input 1

```
4 0 0 0 1
1 10 2 9
```

Sample Output 1

```
11
```

Sample Input 2

```
5 156 125 493 201
101818827 550809474 17607798 702981427 349052006
```

Sample Output 2

```
342962899286
```

Illustrations

- Initially, both elevators are on the **1st floor**.
- In the first test case, the optimal sequence of actions is as follows:

  $F_1 = 1$: Both elevators are already on the 1st floor, so no movement is needed. Cost: 0

  $F_2 = 10$: Move one of the elevators to the 10th floor. Cost: $|1 - 10| = 9$

  $F_3 = 2$: Move the elevator that remained on the 1st floor to the 2nd floor. Cost: $|1 - 2| = 1$

$F_4 = 9$: Move the elevator that is on the 10th floor to the 9th floor. Cost: $|10 - 9| = 1$

Total cost: $0 + 9 + 1 + 1 = 11$

6_四皇后問題(Four Queens Puzzle)

(1.5 分 / 4.5 分 / 9 分)

時間限制: 6.0 second
記憶體限制: 1024 MB

題目敘述

皇后是西洋棋中非常強大的棋子，它可以沿著八種方向在方格上移動到任意格子直到被棋子或邊界擋住，
如果受到阻擋的對象是一個對手的棋子，那皇后可以移動到那個格子並吃掉他。
正式來說，一枚在格子 (x, y) 的皇后可以沿著
$(1, 0), (1, 1), (0, 1), (-1, 1), (-1, 0), (-1, -1), (0, -1), (1, -1)$ 的其中一個方向 (a, b) 移動一個整數倍數 d 如果：

1. 格子 $(x + ad, y + bd)$ 是空的或被對面的棋子佔領。
2. 如果 $d > 1$，那對於所有 $1 \le c < d$，格子 $(x + ac, y + bc)$ 是空的。

在這個情況中，我們說格子 $(x + ad, y + bd)$ 在它的**攻擊範圍**內。

你的朋友又輸掉一場西洋棋錦標賽的資格賽，已經是連續第五十次了，
「這沒道理！」他的皇后動都沒動就被吃掉了，現在他破防到相信皇后很強只是大家一直道聽塗說，實際上根本就沒有。

你想要證明他的陰謀論是錯的，但他出了一道謎題給你。
在 $n \times m$ 的棋盤上，有數個阻擋格子的障礙物，你需要放置剛好**四枚皇后**，使得每一對皇后（總共有六對）都在彼此的攻擊範圍內，而且不會被其他皇后或障礙物阻擋。

為什麼是四枚？只是因為是數量最多的可能，不過這對身為西洋棋大師的你不是難題，找出有多少種放置方法可以滿足條件來否決你朋友的陰謀論。

輸入格式

輸入的第一行包含兩個整數 n, m 代表棋盤的高度與寬度。
接下來有 n 行，每行有一個長度為 m 的字串。
所有字串都由 `@` 與 `.` 組成，第 i 列第 j 個字元代表格子 (i, j) 的狀態：`@` 代表被障礙阻擋而 `.` 代表空的格子。

輸出格式

輸出有多少種放置方法可以滿足題目中的條件。

資料範圍

- $1 \le n, m \le 1600$

子任務

- 子任務 1 (1.5 分): $n, m \le 100$
- 子任務 2 (4.5 分): $n, m \le 700$

- 子任務 3 (9 分): 無額外限制

測試範例

輸入範例 1

```
3 4
.@@@
....
...@
```

輸出範例 1

```
2
```

輸入範例 2

```
5 6
@.....
......
....@.
......
@.....
```

輸出範例 2

```
27
```

輸入範例 3

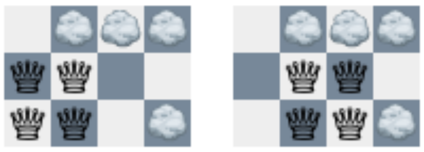
```
6 6
..@...
......
.@....
......
.....@
......
```

輸出範例 3

```
35
```

範例說明

在第一筆範例中，唯二的放置可能是：



6_Four Queens Puzzle

(1.5 points / 4.5 points / 9 points)

Time Limit: 6.0 second
Memory Limit: 1024 MB

Statement

Queens are very powerful pieces in chess.
A queen can move along one of the eight directions on a grid until that direction is blocked.
If a piece of the opposing team blocks it, then the queen can capture it.
Formally speaking, a queen at cell (x, y) can move along one of the eight directions $(1,0),(1,1),(0,1),(-1,1),(-1,0),(-1,-1),(0,-1),(1,-1)$ for any positive integer multiple d if, let (a, b) be the direction chosen:

1. The cell $(x + ad, y + bd)$ is on the board, either not blocked or occupied by another piece.
2. If $d > 1$, for every integer $1 \leq c < d$, the cell $(x + ac, y + bc)$ is empty.

In this case, we say the queen *controls* the cell $(x + ad, y + bd)$.

Your friend just lost another chess tournament... qualifier. Fiftieth in a row.
It is not fair!
His queen had never moved, ever, and got captured.
Now he is so tilted that he is starting to believe an absurd conspiracy theory: the queen is useful only because people say so. It is not.

You want to prove him wrong, but then he challenged you with a puzzle.
Place **four queens** on an $n \times m$ board, with several obstacles scattered around, blocking some of the cells.
Every pair of queens (there are six pairs in total) must mutally control each other's squares, without being blocked by obstacles or other queens.
Every pair of queens (there are six pairs in total) must be able to attack each other, as if they were enemies, with no obstacles or other queens in between.

Why four, you may ask? Because that is simply the maximum possible.
Nevertheless, for a chess master like you, this should be easy.
Count the number of such configurations to prove your friend wrong.

Input Format

The first line of the input contains two integers n and m representing the height and width of the board.
Next, n lines follow, each containing a string of length m.
Each string consists of `@` and `.`.
In the i-th row, the j-th character denotes the status of the cell (i, j) on the board: `@` for an obstacle and `.` for an empty cell.

Output Format

Output the total number of configurations satisfying the conditions in the description.

Constraints

- $1 \le n, m \le 1600$

Subtasks

- Subtask 1 (1.5 points): $n, m \le 100$
- Subtask 2 (4.5 points): $n, m \le 700$
- Subtask 3 (9 points): No additional constraints

Samples

Sample Input 1

```
3 4
.@@@
....
...@
```

Sample Output 1

```
2
```

Sample Input 2

```
5 6
@.....
......
....@.
......
@.....
```

Sample Output 2

```
27
```

Sample Input 3

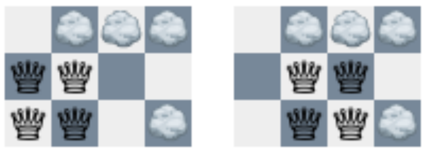
```
6 6
..@...
......
.@....
......
.....@
......
```

Sample Output 3

```
35
```

Illustrations

In the first sample, the two possible configurations are:



7_區間考拉茲操作(Range Collatz Operation)

(6 分 / 14 分)

時間限制: 4.0 second
記憶體限制: 1024 MB

題目敘述

考慮以下定義在正整數上的函數：

$$\text{collatz}(n) = \begin{cases} 3n+1 & \text{if } n \text{ is odd,} \\ n/2 & \text{otherwise.} \end{cases} \tag{2}$$

1937 年，Lothar Collatz 提出了一個猜想：對於任何正整數，重複代入 $\text{collatz}(\cdot)$ 總有一刻會達到 1。

目前已知這個敘述對於所有小於 2.36×10^{21} 的整數都成立。但是，這個猜想是否對所有正整數都成立，仍然是未知數。這些年來，不乏有數學家嘗試證明這個猜想，但都以失敗告終。著名的數學家 Paul Erdős 曾針對此問題給出以下評價：

> "Mathematics may not be ready for such problems."

背景故事就到此為止，以下是真正的問題。

給定一個 n 個正整數的陣列 $a_1, a_2, \ldots, a_n$。

接下來依序進行 q 次操作，每個操作都是以下兩種類型之一。

- $1\ l\ r$: 對於所有 $l \le i \le r$，將 a_i 替換為 $\text{collatz}(a_i)$。
- $2\ l\ r$: 輸出 $\sum_{i=l}^{r} a_i$。

你能在不使用 LLM 的情況下解決這個問題嗎？

輸入格式

第一行輸入兩個整數 n, q。

第二行輸入 n 個正整數 $a_1, a_2, \ldots, a_n$。

接下來輸入 q 行，每一行輸入三個整數 t, l, r，其中第 i 行代表第 i 次操作。

輸出格式

每個第二種操作輸出一行，這行有一個整數代表那次操作的答案。

資料範圍

- $1 \le n, q \le 2 \times 10^5$
- $1 \le a_i \le 10^7$
- $t \in \{1, 2\}$
- $1 \le l \le r \le n$

子任務

- 子任務 1 (6 分): $a_i \leq 26$
- 子任務 2 (14 分): 無額外限制

測試範例

輸入範例 1

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

輸出範例 1

```
29
32
19
17
39
```

7_Range Collatz Operation

(6 points / 14 points)

Time Limit: 4.0 second
Memory Limit: 1024 MB

Statement

Consider the following function defined on positive integers:

$$\text{collatz}(n) = \begin{cases} 3n+1 & \text{if } n \text{ is odd,} \\ n/2 & \text{otherwise.} \end{cases} \tag{3}$$

In 1937, Lothar Collatz proposed a conjecture: for any positive integer, repeatedly applying $\text{collatz}(\cdot)$ will eventually reach 1.

It is currently known that this statement holds true for all integers less than 2.36×10^{21}. However, whether the conjecture holds for all positive integers remains unknown. Over the years, many mathematicians have attempted to prove this conjecture, but all efforts have ended in failure. The renowned mathematician Paul Erdős once remarked on this problem:

> "Mathematics may not be ready for such problems."

Enough with the background story—here is the actual problem.

You are given an array of n positive integers $a_1, a_2, \ldots, a_n$.

You will then perform q operations, each of which is one of the following two types:

$1\ l\ r$: For every $l \le i \le r$, replace a_i with $\text{collatz}(a_i)$.

$2\ l\ r$: Output $\sum_{i=l}^{r} a_i$.

Can you solve this problem without using any LLMs?

Input Format

The first line contains two integers n, q.

The second line contains n positive integers $a_1, a_2, \ldots, a_n$.

Then, q lines follow. Each line contains three integers t, l, r, where the i-th line represents the i-th operation.

Output Format

For each type 2 query, output one line consisting of one integer representing the answer to that query.

Constraints

- $1 \le n, q \le 2 \times 10^5$
- $1 \le a_i \le 10^7$

- $t \in \{1, 2\}$
- $1 \leq l \leq r \leq n$

Subtasks

- Subtask 1 (6 points): $a_i \leq 26$
- Subtask 2 (14 points): No additional constraints

Samples

Sample Input 1

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

Sample Output 1

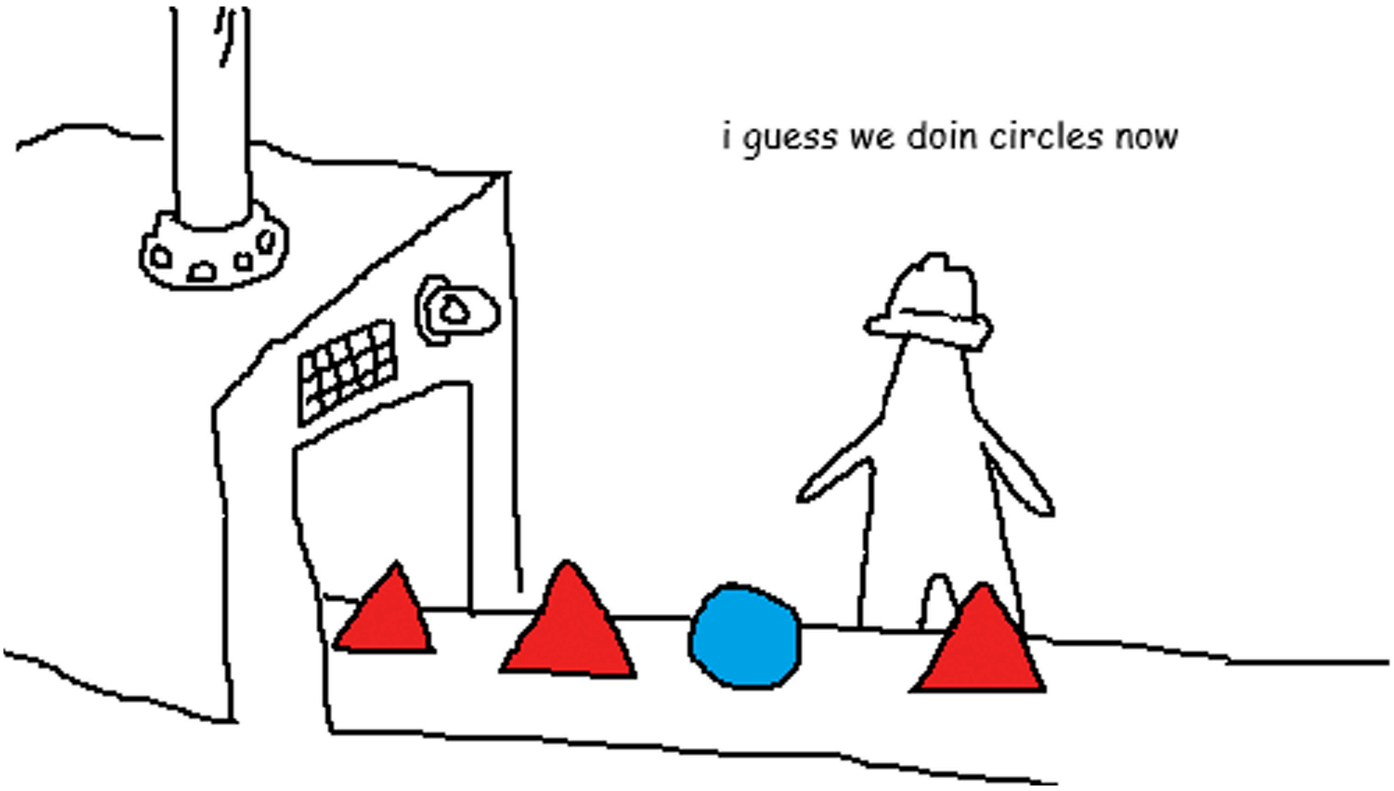
```
29
32
19
17
39
```

8_異常原料偵測(Anomaly Detection)

(2 分 / 4 分 / 6 分 / 8 分)

時間限制: 2.0 second
記憶體限制: 1024 MB

題目敘述



你是三角形工廠的一名員工，最近你們的工廠時不時會製造出奇怪的形狀，為了保障工廠的安全與名聲，你打算仔細調查異常事件出現的原因。

工廠產出的三角形是由正整數所組成的，每一個三角形都可以被一個正整數 a_i 所表示，而我們說這個三角形的組成原料是 **除了 1 和自己以外的正因數**，精確的來說，一個三角形 a_i 的原料集合為

$$S = \{d \mid 1 < d < a_i, d | a_i\}$$

同時**每個三角形的原料中至少存在兩種以上的質數**，也就是說 a_i 至少有兩種質因數。

在追查三角形的原料時，你發現工廠發生異常的原因和一些異常的數字有關係，如果這些數字出現太多次，工廠就會故障，具體來說，在生產的三角形中，如果**有種原料佔據所有原料個數的一半（含）以上**，那麼工廠就會開始發生異常。

為了偵測工廠發生異常的情況，你想要能夠回答異常的原料是否有出現在一堆三角形之中，並將這些異常的原料找出來，也就是說，一開始先給定一堆三角形，你想要能夠維護下列操作：

- 給定一個三角形 k_i，將其加入到三角形堆之中
- 將一個三角形 k_i 於三角形堆中移除（保證能夠在三角形堆中找到 k_i，若有多個 k_i 只移除一個）

每次操作結束後，輸出任意一種異常原料，若沒有則輸出 -1

保證在過程中三角形堆裡永遠存在兩種以上的三角形。

輸入格式

第一行輸入有兩個正整數 n, q，表示一開始三角形堆的大小和操作數量。

第二行有 n 個正整數 $a_1, a_2 \cdots a_n$ 表示一開始三角形堆的樣子。

接下來 q 行，每行包含兩個正整數 t_i, k_i

- 若 $t_i = 1$，表示此操作為將 k_i 加入三角形堆中
- 若 $t_i = 2$，表示此操作為將一個三角形 k_i 從三角形堆中移除。

在二種操作中，保證要刪除的三角形一定存在。

輸出格式

輸出 q 個數字，每個操作之後輸出三角形堆中的任意一種異常原料，若沒有則輸出 -1。

資料範圍

- $2 \leq n \leq 2 \times 10^5$
- $1 \leq q \leq 2 \times 10^5$
- $1 \leq a_i, k_i \leq 10^{18}$ 保證 a_i, k_i 都至少有兩種質因數
- 保證在操作中三角形堆中至少有兩種數字

子任務

- 子任務 1 (2 分): $t_i = 1$
- 子任務 2 (4 分): $1 \leq a_i, k_i \leq 10^6$
- 子任務 3 (6 分): $1 \leq a_i, k_i \leq 10^{12}$
- 子任務 4 (8 分): 無額外限制

測試範例

輸入範例 1

```
3 3
6 10 14
1 14
1 12
2 12
```

輸出範例 1

```
2
-1
2
```

輸入範例 2

```
2 5
62 87
1 115
1 143
2 87
1 133
2 62
```

輸出範例 2

```
-1
-1
-1
-1
-1
```

輸入範例 3

```
2 5
21 33
1 6
1 15
1 33
1 45
1 14
```

輸出範例 3

```
3
3
3
-1
-1
```

範例說明

在第一筆測試資料中，一開始的三角形堆中的原料為：

$\{2, 3, 2, 5, 2, 7\}$

在經過第一個操作後變成

$\{2, 3, 2, 5, 2, 7, 2, 7\}$

2 出現 4 次，所以他是異常原料，輸出 2

在經過第二個操作後變成

$\{2, 3, 2, 5, 2, 7, 2, 7, 2, 3, 6\}$

沒有原料佔據一半或以上，輸出 -1

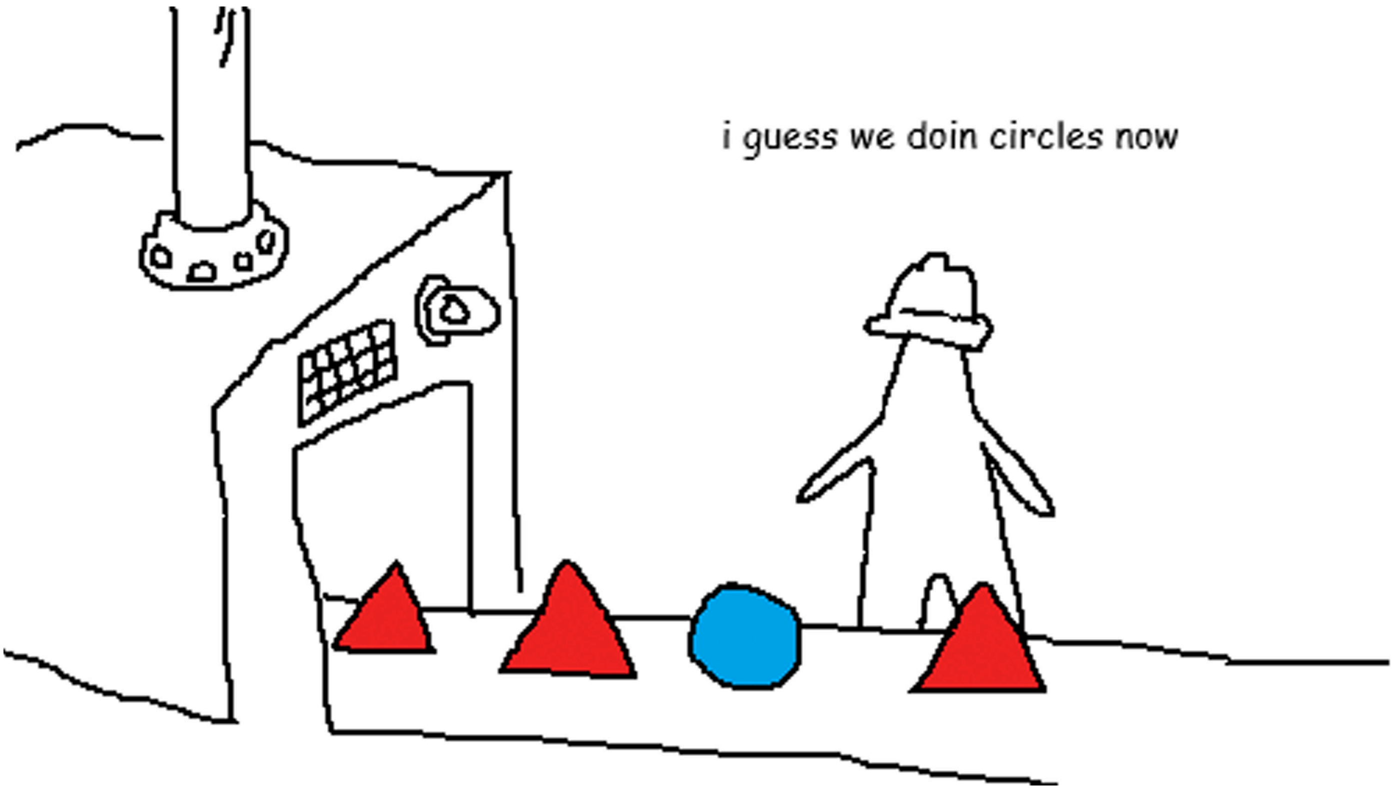
其他以此類推

8_Anomaly Detection

(2 points / 4 points / 6 points / 8 points)

Time Limit: 2.0 second
Memory Limit: 1024 MB

Statement



As a worker in the triangle factory, you've recently noticed the production of some unusual shapes. To ensure the factory's safety and maintain its integrity, you've decided to investigate the cause of these anomalies.

Each triangle produced by the factory is represented by a **positive integer** a_i. We define a triangle's ingredients as **its positive factors, excluding 1 and itself**. More formally, the set of ingredients for a triangle a_i is given by:

$$S = \{d \mid 1 < d < a_i, d|a_i\}$$

It's known that the ingredients of every triangle manufactured by the factory contain at least two distinct prime factors. In other words, each a_i must have at least two prime factors.

Upon investigating the triangles' ingredients, you discovered that the anomalies are linked to certain "weird numbers." If one of these numbers appears too frequently as an ingredient, the factory malfunctions. Specifically, a malfunction occurs if a single ingredient constitutes **more than or equal to half of all ingredients** present in a given pile of triangles.

To detect these anomalies, you need a system to monitor the ingredients within a pile of triangles. You're looking for a way to identify if such an "anomaly ingredient" exists in a pile, and if so, to determine what that ingredient is. That is, given a initial pile of triangles, you are trying to maintain the following operations:

- Given a triangle k_i, add it to the pile of triangles.
- Given a triangle k_i, remove one instance of it from the pile. (It's guaranteed that k_i will be present in the pile when this operation is performed.)

After every operation, you should output one anomaly ingredient. If no such ingredient exists in the pile, output `-1`.

It is guaranteed that, throughout the entire process, the pile will always contain at least two different kinds of triangles.

Input Format

The first line of input contains two positive integers, n and q, representing the initial size of the triangle pile and the number of operations, respectively.

The second line contains n positive integers $a_1, a_2, \ldots, a_n$, describing the initial state of the triangle pile.

The next q lines each contain two positive integers, t_i and k_i:

- If $t_i = 1$, this operation means adding k_i to the pile of triangles.
- If $t_i = 2$, this operation means removing one triangle k_i from the pile.

In the second type of operation, the triangle being remove is guaranteed to be in the pile.

Output Format

Output q intergers, the anomaly ingredient after each operation, if there isn't one, output -1

Constraints

- $2 \leq n \leq 2 \times 10^5$
- $1 \leq q \leq 2 \times 10^5$
- $1 \leq a_i, k_i \leq 10^{18}$, a_i, k_i have at least 2 prime factors.
- It is guaranteed that, throughout the entire process, the pile will always contain at least two different kinds of triangles.

Subtasks

- Subtask 1 (2 points): $t_i = 1$
- Subtask 2 (4 points): $1 \leq a_i, k_i \leq 10^6$
- Subtask 3 (6 points): $1 \leq a_i, k_i \leq 10^{12}$
- Subtask 4 (8 points): No additional constraints

Samples

Sample Input 1

```
3 3
6 10 14
1 14
1 12
2 12
```

Sample Output 1

```
2
-1
2
```

Sample Input 2

```
2 5
62 87
1 115
1 143
2 87
1 133
2 62
```

Sample Output 2

```
-1
-1
-1
-1
-1
```

Sample Input 3

```
2 5
21 33
1 6
1 15
1 33
1 45
1 14
```

Sample Output 3

```
3
3
3
-1
-1
```

Illustrations

In the first test case, the initial ingredients in the triangle pile are:

$\{2, 3, 2, 5, 2, 7\}$

After the first operation, it becomes:

$\{2, 3, 2, 5, 2, 7, 2, 7\}$

Here, 2 appears 4 times, so it's the anomaly ingredient. Output **2**.

After the second operation, it becomes:

$\{2, 3, 2, 5, 2, 7, 2, 7, 2, 3, 6\}$

No single ingredient makes up half or more of the total. Output **-1**.

Other operations follows.