

NSSPC Final 2024

National Taiwan University

2024/09/16

Language	Version	Compile Flags	Extensions
C	gcc 11.3.0	-O2 -x c -static -lm	.c
C++	g++ 11.3.0	-O2 -std=gnu++20 -static	.cc, .cpp, .cxx, .c++
Java	openjdk 17.0.12	[Omitted]	.java

Problem	Problem Name	Points	Time Limit	Memory Limit
A	Nice NSSPC Final	30	1 s	512 MB
B	工作時間	5	1 s	512 MB
C	棒球聯賽	10	1 s	512 MB
D	字串中的字串	15	1 s	512 MB
E	評分	15	1 s	512 MB
F	題目測試	15	1 s	512 MB
G	Star Battle Puzzle	15	1 s	512 MB
H	修復城堡	15	1 s	512 MB
I	表格操作	20	1 s	512 MB
J	種樹	20	1 s	512 MB
K	魔術方塊	30	1 s	512 MB
L	火把製造者	30	1 s	512 MB
M	實驗數據	30	1 s	512 MB
N	整理桌面	40	1 s	512 MB
O	小愛的火柴棒	40	1 s	512 MB



This page is intentionally left blank.

A. Nice NSSPC Final

Problem ID: nice



恭喜晉級 NSSPC 2024 final！比賽已經開始，就讓我們先輸出一行「Nice, NSSPC 2024 Final!」做為暖身吧！

Input

本題沒有輸入。

Output

輸出一行 Nice, NSSPC 2024 Final!

Sample Input 1

Sample Output 1

	Nice, NSSPC 2024 Final!
--	-------------------------



This page is intentionally left blank.

B. 工作時間

Problem ID: work



WiwiHo 很在乎時間管理，她喜歡持續工作 X 小時後休息 Y 小時，休息結束時再馬上開始工作。舉例來說，如果 $X = 5, Y = 2$ ，那麼 WiwiHo 會不間斷地工作 5 小時再休息 2 小時。

假設一開始才剛要開始工作，WiwiHo 在 Z 小時內總共會工作幾個小時？

Input

輸入只有一行，這行包含三個整數 X, Y, Z 。

- $1 \leq X, Y \leq 100$
- $1 \leq Z \leq 10^9$

Output

輸出一個整數，代表 WiwiHo 總共工作了幾個小時。

Sample Input 1	Sample Output 1
5 2 20	15
Sample Input 2	Sample Output 2
99 1 24	24



This page is intentionally left blank.

C. 棒球聯賽

Problem ID: league



番薯國在每年的三月到十月都會舉辦棒球聯賽。聯賽一共有六支球隊，分別以大寫英文字母 A, B, C, D, E, F 來表示。該聯賽會依序進行 N 個事件，事件總共有兩種：

- 比賽事件：兩支不同的球隊進行了一場棒球比賽，並且恰有一方獲得勝利。
- 計分事件：目前勝率最高的隊伍將會獲得 1 分的積分，若有多支隊伍勝率相同則**不會有任何一隊獲得積分**。勝率的計算方法為該隊伍的獲勝數量除以該隊伍的比賽數量，保證該事件發生時 6 隻隊伍都比完了至少一場比賽。

現在按照時間順序給定所有的事件，請計算每隻球隊分別獲得多少積分。

輸入格式

輸入第一行有一個正整數 N ，代表事件的數量。

接下來 N 行，第 i 行代表第 i 個發生的事件。

若第 i 個事件為比賽，則第 i 行會有三個字串 t_i, x_i, y_i 且 $t_i = \text{match}$ ， x_i, y_i 為比賽的隊伍，而 x_i 為獲勝的隊伍。

若第 i 個事件為計分，則第 i 行會有一個字串 t_i 且 $t_i = \text{point}$ 。

- $1 \leq N \leq 1000$
- $t_i \in \{\text{match}, \text{point}\}$
- $x_i, y_i \in \{A, B, C, D, E, F\}$
- $x_i \neq y_i$

輸出格式

請輸出一行，該行有 6 個整數，依序代表 A, B, C, D, E, F 的積分數量。

**Sample Input 1****Sample Output 1**

11 match A D match B E match C F point match A B match B C point match C A point match B E point	1 1 0 0 0 0
---	-------------

Sample Input 2**Sample Output 2**

13 match A C match B D match A E match B F point match A D match B C point match A B match B A point match A D point	1 0 0 0 0 0
---	-------------

D. 字串中的字串

Problem ID: strings



山姆有兩個由小寫英文字母構成的字串 s, t ，他想知道兩個字串是不是相似的。他認為兩個字串相似，若兩字串間存在**非空**共同子字串。

因為字串的長度有可能很長，所以山姆希望你能幫他找出任意一個 s, t 的**非空**共同子字串。

字串 b 是字串 a 的子字串若 b 連續的出現在 a 中。字串 c 是字串 a, b 的非空共同子字串若 c 不是空字串而且 c 既是 a 的子字串，也是 b 的子字串。

Input

輸入有兩行，第一行輸入字串 s ，而第二行輸入字串 t 。

- $1 \leq |s| \leq 10^6$ ，其中 $|s|$ 代表 s 的長度。
- $1 \leq |t| \leq 10^6$ ，其中 $|t|$ 代表 t 的長度。
- 所有字元皆為小寫英文字母 (a 到 z)。

Output

如果不存在任何非空共同子字串，則輸出一行「NO」（不含引號）。

否則輸出兩行。第一行輸出「YES」（不含引號），而第二行則輸出任意一個 s, t 的非空共同子字串。如果有不只一個非空共同子字串，請輸出任意一個。

Sample Input 1

looks good	YES oo
---------------	-----------

Sample Output 1

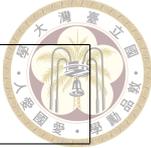
Sample Input 2

hello nsspc	NO
----------------	----

Sample Output 2

Sample Input 3

Sample Output 3



abcdefghijklmnopqrstuvwxy
abcdefghijklmnopqrstuvwxy

YES
abcdefghijklmnopqrstuvwxy

E. 評分

Problem ID: grading



李教授開設了一門叫作「進階區間資料結構」的課程，畢竟課程名稱裡有著「進階」這兩個字，要上這門課的學生當然不是想要上就可以上的，而是要通過李教授的考驗，才能獲准成為這門課的學生。

為了篩選有能力修這門課的學生，李教授出了一道題目，這道題目的正確答案是 $1 \leq N \leq 20$ 個字串 s_1, s_2, \dots, s_N ，其中的每個字串 s_i 都由大小寫英文字母組成，長度在 1 到 20 之間。學生繳交的答案，只要是以**任意順序**輸出這 N 個字串，就會被視為正確，否則會被視為錯誤。

學生們知道的事情只有，這個題目的正確答案會是 1 到 20 個由大小寫英文字母構成、每個長度在 1 到 20 之間的字串。正式地說，一個學生繳交的答案會包含一個整數 M ，代表他認為答案是幾個字串，還有 M 個字串 t_1, t_2, \dots, t_M ，代表他答案中的字串。他的答案會被視為正確的條件是， $N = M$ 且存在一個 1 到 N 的排列 p_1, p_2, \dots, p_N ，滿足 $\forall 1 \leq i \leq N, s_i = t_{p_i}$ ，白話地說就是可以把 t_1, t_2, \dots, t_M 調換順序後變成 s_1, s_2, \dots, s_N 。1 到 N 的排列的意思是 $1, 2, \dots, N$ 各出現恰好一次的序列。

即便這門課的難度相當地高，但還是有許多學生希望能修習這門課程。有 T 個學生提交了他們對這個題目的答案，請你告訴李教授，每一個學生的答案是否正確。

Input

第一行有一個整數 N ，代表正確答案中有幾個字串。

第二行有 N 個以空白分隔的字串 s_1, s_2, \dots, s_N ，代表正確答案中的 N 個字串。

第三行有一個整數 T ，代表有幾個學生提交了他們的答案。

接下來有 T 行，其中的第 i 行代表第 i 個學生提交的答案，這行的一開始是一個整數 M ，代表這名學生認為答案包含幾個字串，接下來有 M 個字串 t_1, t_2, \dots, t_M ，代表他答案中的字串，這 1 個整數和 M 個字串之間以空白分隔。

- $1 \leq N, M \leq 20$
- $1 \leq T \leq 10000$
- 每個 s_i, t_i 都是由大小寫英文字母構成、長度在 1 到 20 之間的字串

Output

輸出 T 行，其中第 i 行，如果第 i 個學生的答案正確，輸出 Yes，否則輸出 No。

Sample Input 1	Sample Output 1
5 good luck and have fun 4 3 luck and fun 5 have fun and good luck 6 good luck and and have fun 5 good luck and have fun	No Yes No Yes
Sample Input 2	Sample Output 2
3 Welcome to NSSPC 5 3 WELCOME TO NSSPC 3 to Welcome NSSPC 2 Welcome NSSPC 1 Welcome 3 welcome to nsspc	No Yes No No No
Sample Input 3	Sample Output 3
6 to be or not to be 3 4 to be or not 6 to to be be or not 9 to be or not to be or to be	No Yes No

F. 題目測試

Problem ID: testers



NSSPC 2024 即將來臨，裁判們必須盡快把題目準備好。裁判團隊有 N 個人，他們打算要出總共 N 道題目，這 N 道題目從 1 到 N 編號，題目 i 的出題者是裁判 i 。

為了確保題目品質，每道題目都要由出題者之外的另一名裁判負責測試，確定題目沒有問題。仔細地說，每一個裁判都要負責測試恰好一道題目，且每道題目都要恰有一個裁判負責測試，這個負責測試的裁判不能和出題者是同一個人。已經有一些裁判先選定了他們要負責測試的題目了，因為出題期限將近，所以你必須盡快為剩下的裁判分配他們要負責測試的題目。

Input

第一行包含一個整數 T ，代表接下來測試資料的組數。

每一筆測試資料的第一行包含一個整數 N ，代表裁判的人數，同時也是要出的題目的數量。

第二行包含 N 個整數 a_1, a_2, \dots, a_N 。對於 $1 \leq i \leq N$ ，如果 $a_i \neq -1$ ，代表裁判 i 已經選定了要負責測試題目 a_i ；如果 $a_i = -1$ ，代表裁判 i 還沒選擇要負責測試哪道題目。

- $1 \leq T$
- $2 \leq N \leq 10^5$
- T 筆測試資料中的 N 總和 $\leq 10^5$
- $1 \leq a_i \leq N$ 或 $a_i = -1$
- 保證有辦法為所有 $a_i = -1$ 的裁判指定負責測試的題目，滿足所有要求

Output

對於每筆測試資料，輸出一行，包含 N 個整數 b_1, b_2, \dots, b_N ，其中 $1 \leq b_i \leq N$ ，代表你分配之後，裁判 i 負責測試題目 b_i 。如果本來一個裁判已經選定要測試哪一題，你必須保留他本來選定的題目。正式地說，你輸出的答案必須滿足以下所有條件：

- $1 \leq b_i \leq N$
- $b_i \neq i$
- 如果 $a_i \neq -1$ ，那麼 $b_i = a_i$

- 對於 $1 \leq j \leq N$ ，恰有一個 i 滿足 $b_i = j$

**Sample Input 1**

```
5
5
-1 -1 -1 -1 -1
5
3 4 2 5 1
3
3 -1 -1
3
-1 -1 1
10
3 -1 -1 5 2 -1 1 6 -1 4
```

Sample Output 1

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

G. Star Battle Puzzle

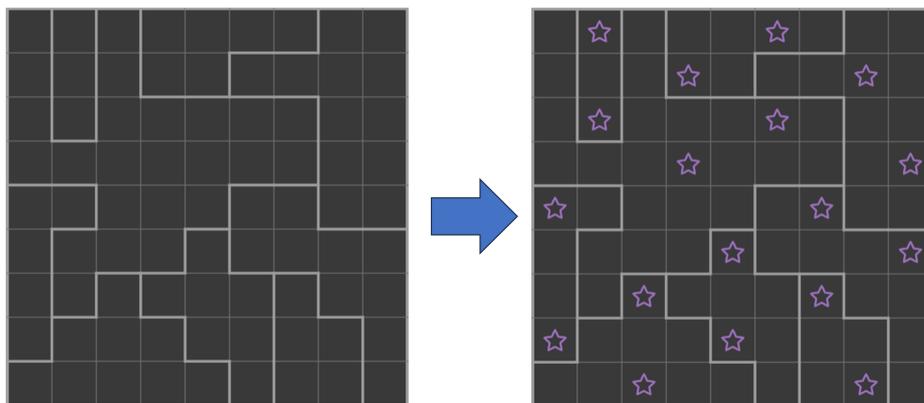
Problem ID: star-battle



你有玩過 Star Battle Puzzle 嗎？Star Battle Puzzle 是一個解謎遊戲，每一道謎題都會以一個空的 9×9 盤面開始，該盤面會被切割成若干個區塊。玩家的目的是在一些空格填入一些「星星」，滿足：

- 每一行都有恰兩個星星
- 每一列都有恰兩個星星
- 每一個區塊都有恰兩個星星
- 任兩個星星都不得八方位相鄰

下圖是一道空的盤面及其解答的示意圖：



現在，請你撰寫一支程式，在讀入一個已經填好星星的 Star Battle Puzzle 盤面後，輸出該盤面是否有被完整解決。意即填入的星星是否有滿足所有指定條件。

Input

輸入首先會是空盤面的資訊，共有九行，每行九個字元，每個字元皆為數字 1 至 9 其中之一，代表該格隸屬於的區塊編號。

緊接著是填入星星的方式，共有九行，每行九個字元，每個字元 . 或 *，其中 . 代表未填入任何東西、* 代表填入一顆星星。

- 保證給定的空盤面由字元 1 至 9 組成

- 保證給定的空盤面的共有九個區塊，且每個區塊四方位連通
- 保證給定的填入星星方式由字元 . 和 * 組成



Output

若輸入的填入星星方式滿足所有條件，輸出 Valid；否則只要違反任何一個條件，輸出 Invalid。

Sample Input 1	Sample Output 1
<pre> 121333344 121334444 121111144 111111144 551116644 511176666 518777966 588877996 888887996 .*...*... ...*...*. .*...*... ...*...*. *...*. ...*...* ..*...*.. .*...*.. *...*... ..*...*.. </pre>	<pre> Valid </pre>



Sample Input 2	Sample Output 2
<pre> 121333344 121334444 121111144 111111144 551116644 511176666 518777966 588877996 888887996 .*...*... ...*...*. .*.....* ...*.*. *.....*.. ...*.*. .*...*.. *...*... ..*...*. </pre>	<p>Invalid</p>

Sample Input 3	Sample Output 3
<pre> 121333344 121334444 121111144 111111144 551116644 511176666 518777966 588877996 888887996 .*..... ...*...*. .*...*... ...*...*. *.....*.. ...*.*. .*...*.. *...*... ..*...*. </pre>	<p>Invalid</p>



This page is intentionally left blank.

H. 修復城堡

Problem ID: repair



小波有一座城堡，他每天都在他的城堡裡過著快樂的日子，直到有一天，他的城堡被小奕攻擊了。在小奕攻擊小波的城堡時，城堡的某一面城牆居然被小奕完全摧毀了！剛好小波覺得這面城牆本來就長得很醜，正好可以重新建造一次這面城牆。

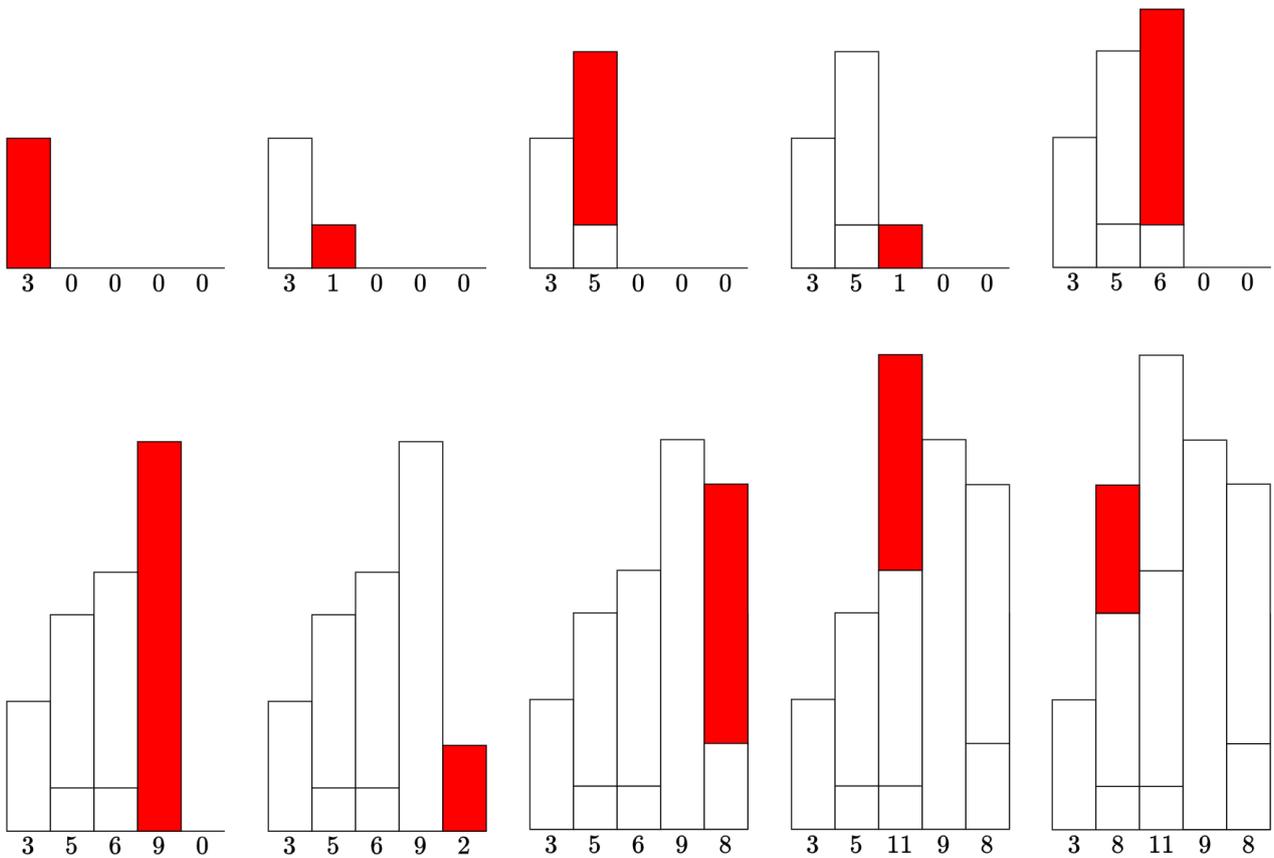
這面城牆的寬度是 N 公尺，他將整面城牆劃分成 N 個寬 1 公尺的區塊，由左至右從 1 到 N 編號，並且訂購了 Q 個寬度為 1、厚度和城牆一樣、高度不等的石磚，一個石磚必須剛好放進一個區塊之中，不能跨越多個區塊，也不能旋轉。正式地說，這 N 個區塊有各自的高度，一開始高度都是 0，如果一個區塊本來的高度是 x ，接著小波在這個區塊多放一個高度為 y 的石磚，在這之後這個區塊的高度會變成 $x + y$ 。

小波訂購的 Q 個石磚會依序送達，第 i 個石磚的高度是 h_i ，然而小波在拿到石磚之前，都不會知道他訂購的石磚高度究竟是多少，他又迫不及待地想要看到城牆蓋好的樣子，因此他每收到一個石磚，就要立刻選一個區塊來放。他選擇區塊的原則是這樣的：如果目前第 i 個區塊的高度是 x_i ，那這面城牆的醜度就是

$$\sum_{k=0}^N (x_k - x_{k+1})^2$$

特別地， $x_0 = x_{N+1} = 0$ 。小波希望選擇一個區塊放置這個石磚，使得放下去之後，這面城牆的醜度盡量小。如果有多個區塊都會讓醜度最小，他會選擇編號最小的區塊。

舉例來說， $N = 5$ 、 $Q = 10$ ，10 個石磚的高度依序是 3, 1, 4, 1, 5, 9, 2, 6, 5, 3。在放第一個石磚的時候，無論放在哪裡，放下後的醜度都會是 18，因此小波會將第一個石磚放在第 1 個區塊。在放第二個石磚的時候，放在區塊 1 的話醜度會是 32、放在區塊 2 的話醜度會是 14、放在其他區塊則會是 20，因此小波會將第二個石磚放在區塊 2。過程中每個區塊的高度變化如下圖，紅色的石磚代表剛放下去的石磚，下方的數字代表該區塊的高度。



小奕偷偷知道了這 Q 個石磚的高度，他很好奇小波新蓋的城牆會長什麼樣子，以便規劃下一次的攻城行動，請你告訴小奕最終每個區塊的高度會是多少。

Input

第一行有一個整數 N, Q ，分別代表城牆的寬度與石磚的數量。

第二行有 Q 個以空白間隔的整數 h_1, h_2, \dots, h_Q ，代表 Q 個依序送達的石磚的高度。

- $1 \leq N, Q \leq 500$
- $1 \leq h_i \leq 10^5$

Output

輸出 N 個以空白間隔的整數 x_1, x_2, \dots, x_N ，代表第 i 個區塊最終的高度是 x_i 。



Sample Input 1

5 10 3 1 4 1 5 9 2 6 5 3	3 8 11 9 8
-----------------------------	------------

Sample Output 1

Sample Input 2

3 4 100000 100000 100000 100000	100000 200000 100000
------------------------------------	----------------------

Sample Output 2



This page is intentionally left blank.

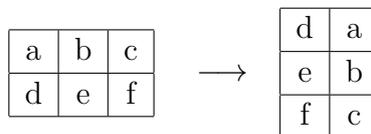
I. 表格操作

Problem ID: operations



現在有一個 $N \times M$ 表格，每個格子內有一個小寫英文字母，我們用 (i, j) 代表由上而下第 i 橫列，由左而右第 j 直排的格子。你必須對表格依序執行 Q 次操作，操作主要分成兩種。

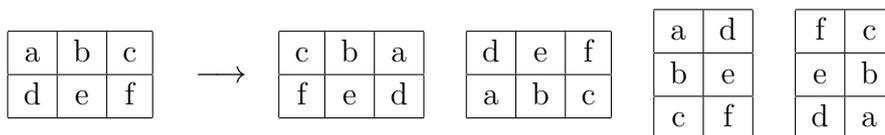
第一種操作為「旋轉」操作，每個旋轉操作會伴隨著一個正整數 t ，表示要連續的將整個表格順時針旋轉 90 度 t 次。一個 $R \times C$ 的表格順時針旋轉 90 度後，會變成一個 $C \times R$ 的表格，而原本位於 (i, j) 的格子會跑到 $(j, R + 1 - i)$ 。舉例來說，以下左邊 2×3 的表格旋轉 90 度一次會變成右邊 3×2 的表格。



第二種操作為「翻轉」操作，這種操作還會再細分成四種翻轉操作：水平翻轉、垂直翻轉以及主副對角線翻轉。這四種翻轉操作的變化如下：

- 水平翻轉：一個 $R \times C$ 的表格經過一次水平翻轉仍然會是一個 $R \times C$ 的表格，但原本在 (i, j) 的格子會跑到 $(i, C + 1 - j)$ 。
- 垂直翻轉：一個 $R \times C$ 的表格經過一次垂直翻轉仍然會是一個 $R \times C$ 的表格，但原本在 (i, j) 的格子會跑到 $(R + 1 - i, j)$ 。
- 主對角線翻轉：一個 $R \times C$ 的表格經過一次主對角線翻轉後會變成一個 $C \times R$ 的表格，而原本在 (i, j) 的格子會跑到 (j, i) 。
- 副對角線翻轉：一個 $R \times C$ 的表格經過一次副對角線翻轉後會變成一個 $C \times R$ 的表格，而原本在 (i, j) 的格子會跑到 $(C + 1 - j, R + 1 - i)$ 。

舉例來說，以下左邊 2×3 的表格經過水平、垂直、主副對角線翻轉後會分別變成右邊四個表格。



執行完所有操作後，請回答最後表格的長相。



Input

第一行輸入三個整數 N, M, Q 。

接下來輸入 N 行，第 i 行輸入一個長度為 M 的字串 s_i ，其中 s_i 的第 j 個字元代表一開始 (i, j) 內的小寫英文字母。

接下來再輸入 Q 行，第 i 行包含一個英文字母 op_i 和一個整數 t_i ，其意義如下：

- 如果 $op_i = R$ ，則代表第 i 次操作為旋轉操作，且要連續旋轉 90 度 t_i 次。
- 如果 $op_i = F$ ，則代表第 i 次操作為翻轉操作，而 $t_i = 1, 2, 3, 4$ 依序代表該次操作為水平翻轉、垂直翻轉、主對角線翻轉以及副對角線翻轉。

- $1 \leq N, M, Q \leq 100$
- 保證所有 s_i 僅由小寫英文字母組成
- op_i 只可能是 R 或 F
- 如果 $op_i = R$ ，則 $1 \leq t_i \leq 10^9$
- 如果 $op_i = F$ ，則 $1 \leq t_i \leq 4$

Output

第一行輸出兩個正整數 N', M' ，代表經過所有操作後會得到一個 $N' \times M'$ 的表格。

接下來輸出 N' 行，第 i 行輸出一個長度為 M' 的字串 s'_i ，其中 s'_i 的第 j 個字元代表最後 (i, j) 內的小寫英文字母。

Sample Input 1

```
3 10 4
baluteshah
littlecube
hsinmutsai
F 3
R 2
F 1
R 103
```

Sample Output 1

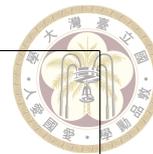
```
3 10
iastumnish
ebuceltil
hihsetulab
```

Sample Input 2

```
3 6 4
wiwiho
howiwi
wihowi
F 4
F 2
R 49
R 1
```

Sample Output 2

```
6 3
oii
hww
iio
wwh
ioi
whw
```





This page is intentionally left blank.

J. 種樹

Problem ID: tree



王老先生有塊地，這塊地可以用一個 $N \times M$ 的表格表示，一開始表格的每個位置都種著一棵高度為 0 的樹。令橫列的編號由上到下，而直排為由左到右，同時令第 i 橫列與第 j 個直排交匯處的格子為 (i, j) 。王老先生有很多個成長劑，他可以選擇一個滿足 $2 \leq i \leq N - 1, 2 \leq j \leq M - 1$ 的格子 (i, j) 並在該格子使用，使用後在所有滿足 $|x - i| \leq 1, |y - j| \leq 1$ 的格子 (x, y) 的樹高度都會增加 1。

今天早上王老先生用了數個生長劑，但到了下午他卻忘記他在哪些位置使用了生長劑。現在王老先生告訴你每棵樹的高度，你能幫幫他還原出他在每個位置使用了生長劑幾次嗎？

輸入格式

輸入第一行有兩個正整數 N, M ，代表表格的大小。

接下來 N 行，第 i 行有 M 個數字 $a_{i,1}, a_{i,2}, \dots, a_{i,M}$ ，其中 $a_{i,j}$ 代表格子 (i, j) 上的樹高度。

- $3 \leq N \leq 500$
- $3 \leq M \leq 500$
- $0 \leq a_{i,j} \leq 10^6$
- 保證王老先生沒有數錯樹的高度，且生長劑為唯一可以增加樹高度的方法

輸出格式

請輸出 $N - 2$ 行，第 i 行有 $M - 2$ 個整數 $b_{i,1}, b_{i,2}, \dots, b_{i,M-2}$ ，其中 $b_{i,j}$ 代表王老先生在格子 $(i + 1, j + 1)$ 使用了幾個生長劑。

Sample Input 1	Sample Output 1
4 5	0 1 2
0 1 3 3 2	1 0 1
1 2 5 4 3	
1 2 5 4 3	
1 1 2 1 1	

Sample Input 2

Sample Output 2



4 3	0
0 0 0	0
0 0 0	
0 0 0	
0 0 0	

K. 魔術方塊

Problem ID: rubiks-cube



FHvirus 一直沈迷於魔術方塊。要知道，認真的投入魔術方塊除了要能夠順暢的解出各種魔術方塊外，還要能解得快。對此，他不但接觸了各種不同種類的魔術方塊，還參加了各種大大小小的魔術方塊比賽。

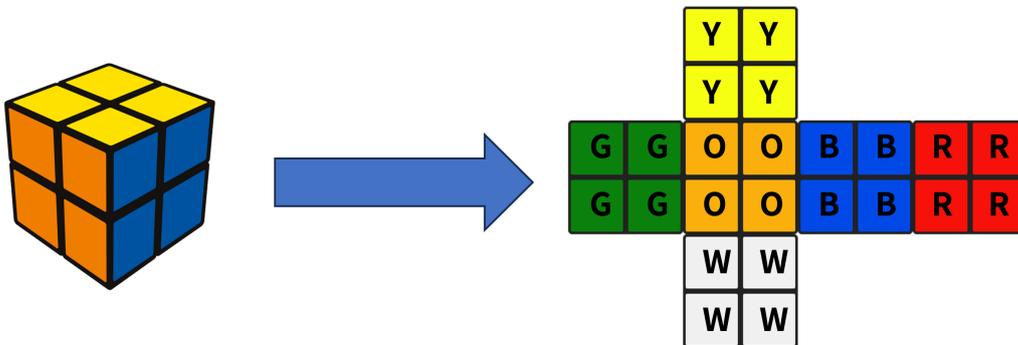
這樣喜愛魔術方塊的 FHvirus 在最近接觸到了程式設計，同時也很喜歡程式設計的他決定結合他在魔術方塊的興趣，自己寫出一個「魔術方塊打亂器」來幫助他練習。

什麼是魔術方塊打亂器呢？魔術方塊玩家為了要練習，他們必須時常重複將魔術方塊打亂再嘗試快速復原的動作，但如果一直隨意的打亂的話，人類無可避免的習慣會讓他們總是接觸到類似的打亂方法。因此，撰寫一個公平的「打亂器」就顯得非常重要了。

因此，所謂的「魔術方塊打亂器」，就是會負責隨機產生一串「打亂公式」，再由玩家照著這個公式打亂。玩家在打亂後，再照著自己習慣的方式練習快速復原。

不過，FHvirus 才剛開始學程式而已，因此他決定先從最簡單的「二階魔術方塊打亂器」開始寫起。

首先我們要認識什麼是「二階魔術方塊」，如下圖：



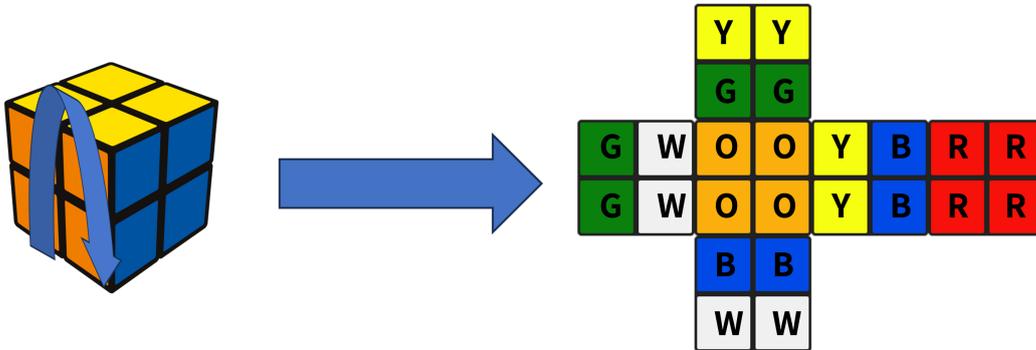
上圖中，左方就是一個二階魔術方塊的樣子，他有著六個面，每個面分別由四格方格組成，常由黃、橘、藍、紅、綠、白六種顏色組成。而右方則是他的展開圖，可以實際看見六個面的顏色，其中黃、橘、藍、紅、綠、白分別以 Y、O、B、R、G、W 簡寫之。

接著，我們會介紹三個在二階魔術方塊上的基本操作。



1. F 操作

F 操作會順時針地轉動魔術方塊前方的面，如下圖：

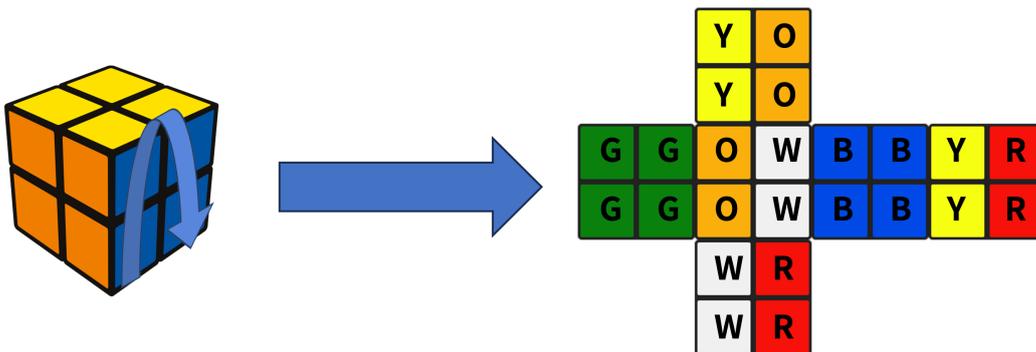


上圖中，左側顯示了旋轉的面及方向，右側則是轉動後展開圖會變成的樣子。

為了簡化步數，我們會以 F^2 則表示「180 度地旋轉魔術方塊前方的面」，同時也可以想成執行 F 操作兩次； F' 來表示「逆時針地轉動魔術方塊前方的面」，同時也可以想成執行 F 操作三次。

2. R 操作

R 操作會順時針地轉動魔術方塊右方的面，如下圖：



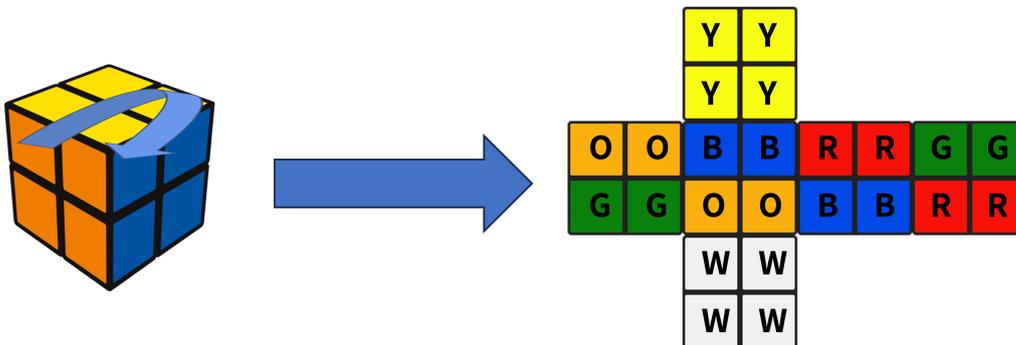
上圖中，左側顯示了旋轉的面及方向，右側則是轉動後展開圖會變成的樣子。

為了簡化步數，我們會以 R^2 則表示「180 度地旋轉魔術方塊右方的面」，同時也可以想成執行 R 操作兩次； R' 來表示「逆時針地轉動魔術方塊右方的面」，同時也可以想成執行 R 操作三次。



3. U 操作

U 操作會順時針地轉動魔術方塊上方的面，如下圖：



上圖中，左側顯示了旋轉的面及方向，右側則是轉動後展開圖會變成的樣子。

為了簡化步數，我們會以 U^2 則表示「180 度地旋轉魔術方塊上方的面」，同時也可以想成執行 U 操作兩次； U' 來表示「逆時針地轉動魔術方塊上方的面」，同時也可以想成執行 U 操作三次。

有了以上這些基本操作後，是時候開始來寫二階魔術方塊打亂器了。首先，我們必須先寫出一份「打亂公式產生器」，是負責產生一連串的基本操作用的。接著，我們還得寫出一份「轉動模擬器」，負責模擬出二階魔術方塊照著打亂公式打亂後的樣子，來讓玩家可以在自行照著公式打亂後，確認自己是否有打亂錯誤。

產生器的部份 `FHVirus` 已經先寫好了，不過模擬器的部份他卻遇到了障礙。你可以幫他撰寫一個模擬器，在讀入打亂公式後，輸出二階魔術方塊照著打亂公式打亂後的樣子嗎？

Input

輸入首行有一個正整數 N ，代表該次產生器產生出來的打亂公式長度。

接下來一行 N 個字串，每個字串會是九種基本操作， F 、 F' 、 F^2 、 R 、 R' 、 R^2 、 L 、 L' 、 L^2 之一。

- $1 \leq N \leq 100$



Output

輸出照著打亂公式打亂後魔術方塊的展開圖，最初的狀態如題敘中第一張圖右方的展開圖所示。

對於展開圖，評測端會進行嚴格比對，意即你必須依照以下範例格式輸出：

```

YY
YY
GGOOBBRR
GGOOBBRR
  WW
  WW

```

其中第一、二、五、六行最前方都有兩個空格字元（ASCII 編號 32），且每一行最後都有一個換行字元（ASCII 編號 10）。注意到第一、二、五、六行的後方都沒有多餘的空格字元，也就是他們的長度比四、五行都要少四。

Sample Input 1

<pre> 8 U F R F R U F R </pre>	<pre> WB RG OBWWORDYB GYGOYGYR RB WO </pre>
--------------------------------	---

Sample Output 1

Sample Input 2

<pre> 10 F2 R' U F' R2 U2 F R' F' R </pre>	<pre> OY RG GBWYROBY GRBBWOWR YO WG </pre>
--	--

Sample Output 2

L. 火把製造者

Problem ID: torch



小方塊喜歡玩方塊遊戲，他最近發現了一款新的方塊遊戲。在這個遊戲中，有五種物品：原木、木材、木炭、木棍和火把。小方塊現在手上有 N 個原木，他很喜歡火把，所以他希望利用這些原木做出盡量多的火把。小方塊要怎麼做出火把呢？他有兩種操作可以做：**合成**和**熔煉**。

- 合成：根據配方消耗需要的材料，就能獲得指定的物品。以下是所有的合成配方，消耗箭頭左邊的物品，就能獲得右邊的物品。

1 個原木 → 4 個木材
 2 個木材 → 4 個木棍
 1 個木棍和 1 個木炭 → 1 個火把

- 熔煉：1 次熔煉可以將 1 個原木變成 1 個木炭，不過必須要消耗 1 點能量。獲得能量的方式是消耗燃料，除了火把以外的物品都可以當作燃料，下表是每種物品當作燃料消耗後可以獲得的能量點數：

燃料	能量
1 個原木	1 點
1 個木材	1 點
1 個木棍	0.2 點
1 個木炭	8 點

注意如果要把一個物品當成燃料消耗，就必須把那整個物品當成燃料消耗掉，例如你不能只消耗 $1/8$ 個木炭獲得 1 點能量。

舉例來說，如果小方塊一開始有 10 個原木，那他可以先把 3 個原木合成成 $3 \times 4 = 12$ 個木材，把其中 7 個木材當成燃料後獲得 7 點能量，用來把剩下的 7 個原木熔煉成木炭，再將剩下 5 個木材的其中 4 個合成成 $\frac{4}{2} \times 4 = 8$ 個木棍，最終用其中 7 個木棍和 7 個原木合成出 7 個火把。

請告訴小方塊，他最多可以做出幾個火把。

Input

第一行包含一個整數 T ，代表測試資料的數量。

接下來有 T 行，每一行是一筆測試資料。一筆測試資料包含一個整數 N ，代表小方塊有幾個原木。



- $1 \leq T \leq 10^5$
- $1 \leq N \leq 10^{12}$

Output

輸出 T 行，其中第 i 行輸出一個整數，代表第 i 筆測試資料的答案。

Sample Input 1

Sample Output 1

14	0
1	1
2	2
3	2
4	3
5	4
6	4
7	5
8	6
9	7
10	78
100	214399693511
271828182845	247787589578
314159265359	788732394365
1000000000000	

M. 實驗數據

Problem ID: experiments



查理這陣子在進行某項研究。這項研究會需要進行若干次實驗，每次實驗會得到一個分數，代表該次實驗的結果。查理一共做了 N 次實驗，第 i 次實驗的分數為 a_i 。

查理接下來要開始分析他所得到的數據，分析的過程會需要計算某些區間的變異數。更明確的說，有 Q 組 (l_i, r_i) ，對於每一組 (l_i, r_i) ，查理需要算出 $a_{l_i}, a_{l_i+1}, \dots, a_{r_i}$ 的變異數。

查理不知道該如何快速的算出這 Q 個變異數，於是他來尋求你的協助。請你幫助查理算出他需要的 Q 個變異數。

K 個實數 x_1, x_2, \dots, x_K 的變異數為 $\frac{1}{K} \sum_{i=1}^K (x_i - \bar{x})^2$ ，其中 $\bar{x} = \frac{1}{K} \sum_{i=1}^K x_i$ 。

Input

第一行輸入兩個整數 N, Q 。

第二行輸入 N 個整數 a_1, a_2, \dots, a_N 。

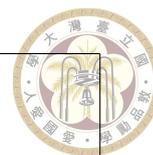
接下來輸入 Q 行，其中第 i 行輸入兩個整數 l_i, r_i 。

- $1 \leq N, Q \leq 2 \cdot 10^5$
- $-10^6 \leq a_i \leq 10^6$
- $1 \leq l_i \leq r_i \leq N$

Output

輸出 Q 行，第 i 行輸出一個整數，代表 $(r_i - l_i + 1)^2 \cdot V_i$ ，其中 V_i 為 $a_{l_i}, a_{l_i+1}, \dots, a_{r_i}$ 的變異數。

可以證明 $(r_i - l_i + 1)^2 \cdot V_i$ 在給定的條件下一定會是整數。

**Sample Input 1**

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

Sample Output 1

```
24
0
50
```

Sample Input 2

```
6 3
-49 49 49 -49 100 -100
1 6
2 3
4 6
```

Sample Output 2

```
177624
0
64802
```

N. 整理桌面

Problem ID: desktop



你是否不小心放太多文件在電腦桌面，導致桌面凌亂不堪的經驗呢？習慣將環境整理乾淨的人們可能沒有，但 8e7 就不是這樣的人。這天，8e7 發現他的電腦桌面實在是太過凌亂，其中一項很大的原因就是因為他留了太多沒有用的文件在桌面上當作暫存，只要將這些文件刪除，桌面勢必可以乾淨不少。

但 8e7 不習慣將環境整理乾淨的原因終究還是因為懶惰，因此，他希望能花最少的力氣來刪除這些文件。8e7 發現，他只要在桌面上選取一個方框，就可以將方框內的文件都變成選取狀態，只要點選刪除，就可以一口氣把選取狀態中的文件全數刪除，不過當他選取第二個方框後就發現事情並沒有那麼單純！

實際上，8e7 電腦的作業系統是這樣設計的：每當 8e7 在桌面上選取一個方框後，所有在方框內的文件選取狀態都會反轉。

什麼意思呢？也就是說，只要在方框內的文件原本是未選取狀態，這份文件就會變成選取狀態；而如果這份文件本來就已經是選取狀態，則他會變回未選取狀態！

這樣的操作變得稍顯複雜許多，但 8e7 仍然不想放棄使用最少的力氣來選取所有的文件好讓他一鍵刪除它們。現在，8e7 已經將所有的文件轉換成 2D 平面上的 N 個座標，其中有 M 個文件是需要被刪除的，請你撰寫一支程式，在讀入所有文件的座標後，給 8e7 一組使用**最少次方框選取操作**框出所有文件的方法。

Input

輸入首行有兩個正整數 N, M ，代表 8e7 電腦桌面總共的文件數和在其中需要被刪除的文件數。

接下來 N 行，第 i 行兩個整數 x_i, y_i ，代表第 i 個文件的座標。其中第 $1 \sim M$ 個文件為 8e7 希望刪除的文件。

- $1 \leq M \leq N \leq 14$
- $-10^9 \leq x_i, y_i \leq 10^9$
- 保證所有文件的座標兩兩相異



Output

首行輸出一個正整數 k ，代表你需要使用的方框選取操作數量。

接下來 k 行，第 j 行要有四個整數 l_j, d_j, r_j, u_j ，代表你第 j 次要選取的方框為座標 (l_j, d_j) 和 (r_j, u_j) 形成的矩形，必須滿足 $-2 \times 10^9 \leq l_j \leq r_j \leq 2 \times 10^9$ 且 $-2 \times 10^9 \leq d_j \leq u_j \leq 2 \times 10^9$ 。代表對於第 i 個文件，若文件 i 滿足 $l_j \leq x_i \leq r_j$ 且 $d_j \leq y_i \leq u_j$ 的話，他的選取狀態就會被反轉。

你輸出的選取方法必須恰讓第 $1 \sim M$ 個文件在選取狀態、其餘的文件則需要在未選取狀態。若有多組可能的解答，輸出任何一種皆會被視為正確。

Sample Input 1

5 3	2
1 4	1 2 5 5
2 2	3 3 4 5
5 4	
3 3	
4 5	

Sample Output 1

Sample Input 2

6 3	3
-3 5	-3 0 6 0
-3 9	-3 9 -3 9
6 0	-3 0 -3 5
6 5	
-3 8	
6 2	

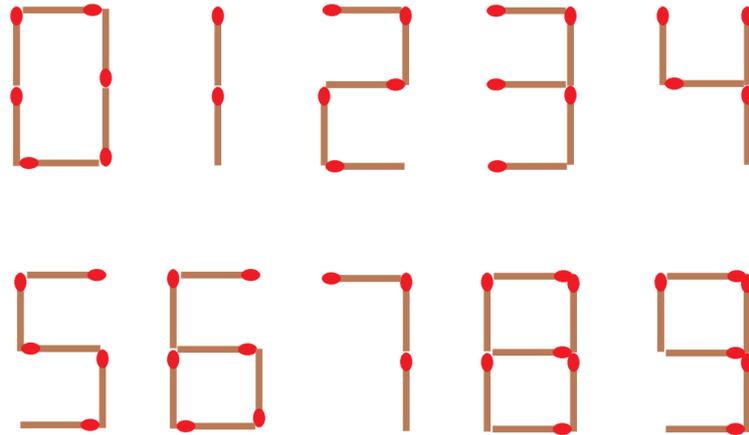
Sample Output 2

O. 小愛的火柴棒

Problem ID: matchsticks



眾所周知，火柴棒可以拿來拼數字，而 0 到 9 每個數字的拼法如下圖所示：



不同的整數可能會需要使用不同數量的火柴棒，例如：17 需要一個 1 和一個 7，所以要拼出 17 總共需要 $2 + 3 = 5$ 支火柴棒，而要拼出 45510 則需要 $4 + 5 + 5 + 2 + 6 = 22$ 支火柴棒。

小愛手上有 N 支火柴棒，他想要用盡所有的火柴棒拼出一個非負整數。

小愛拼出的非負整數能有幾種？由於答案可能很龐大，所以請回答答案除以 998244353 的餘數。

Input

輸入只有一個正整數 N 。

- $1 \leq N \leq 10^{18}$

Output

請輸出一個非負整數，代表小愛能拼出的非負整數種類數除以 998244353 的餘數。



Sample Input 1	Sample Output 1
7	12
Sample Input 2	Sample Output 2
3	1
Sample Input 3	Sample Output 3
1000000	811496257