2023 網際網路程式設計全國大賽
國中組初賽

- 本次比賽共 6 題，含本封面共 20 頁。
- 全部題目的輸入都來自**標準輸入**。輸入中可能包含多組輸入，以題目敘述為主。
- 全部題目的輸出皆輸出到螢幕（**標準輸出**）。

  輸出和題目指定的輸出格式必須完全一致，英文字母大小寫不同或有多餘字元皆視為答題錯誤。
- 比賽中上傳之程式碼，使用 C 語言請用 `.c` 為副檔名；使用 C++ 語言則用 `.cpp` 為副檔名。
- 使用 `cin` 輸入速度遠慢於 `scanf` 輸入，若使用需自行承擔 `TIMELIMIT` 的風險。
- 部分題目有浮點數輸出，會採容許部分誤差的方式進行評測。一般來說「相對或絕對誤差小於 ϵ 皆視為正確」，ϵ 值以題目敘述為主。

  舉例來說，假設 $\epsilon = 10^{-6}$ 且 a 是正確答案，b 是你的答案，如果符合 $\frac{|a-b|}{\max(|a|,|b|,1)} \leq 10^{-6}$，就會被評測程式視為正確。

Problem	Problem Name	Time Limit	Memory Limit
A	逆轉	1 s	1024 MB
B	密碼測試	1 s	1024 MB
C	巫醫巫醫滅滅滅	2 s	1024 MB
D	李杈搭電梯	1 s	1024 MB
E	果樹大盜	1 s	1024 MB
F	烏龜疊疊疊	1 s	1024 MB

2023 網際網路程式設計全國大賽
輸入輸出範例

C 程式範例：

```c
#include <stdio.h>
int main()
{
    int cases;
    scanf("%d", &cases);
    for (int i = 0; i < cases; ++i)
    {
        long long a, b;
        scanf("%lld %lld", &a, &b);
        printf("%lld\n", a + b);
    }
    return 0;
}
```

C++ 程式範例：

```cpp
#include <iostream>
int main()
{
    int cases;
    std::cin >> cases;
    for (int i = 0; i < cases; ++i)
    {
        long long a, b;
        std::cin >> a >> b;
        std::cout << a + b << std::endl;
    }
    return 0;
}
```

A. 逆轉

Problem ID: comeback

一年一度的全國系統程式設計大賽（National Systems Programming Contest，簡稱 NSPC）即將結束！今年的比賽一共有 N 支隊伍參賽，角逐著全國系統程式設計冠軍的稱號。

今年的 NSPC 一共有 M 道題目，比賽時間一共有 300 分鐘。一支隊伍的名次會由他們的「答對題數」、「罰時」及「最後一次答對題目的時間」決定。假設某支隊伍一共答對了 K 題，他們分別在比賽開始後第 $m_1, m_2, \ldots, m_K$ 分鐘解出這些題目，並且各題分別嘗試了 $s_1, s_2, \ldots, s_K$ 次後通過，則：

- 他們的答對題數為 K。
- 他們的罰時為 $(m_1 + m_2 + \cdots + m_K) + [(s_1 - 1) + (s_2 - 1) + \cdots + (s_K - 1)] \times 20$。特別注意，罰時只計算已經通過的題目。未解出的題目嘗試的次數不會造成罰時。
- 若他們的答對題數為 0，則他們的最後一次答對題目的時間為 0；否則他們的最後一次答對題目的時間為 $m_1, m_2, \ldots, m_K$ 中的最大值。

而隊伍之間排名的方式如下：

- 答對題數較多的隊伍排名在前。
- 答對題數相同時，則罰時較低的隊伍排名在前。
- 答對題數及罰時皆相同時，則最後一次答對題目的時間較早的隊伍排名在前。
- 若答對題數、罰時及最後一次答對題目的時間皆相同時，兩支隊伍的排名並列。

舉例來說，假設有三支隊伍 A, B, C 皆答對了 4 題。隊伍 A 於比賽的第 $25, 35, 45, 55$ 分鐘解出這些題目，隊伍 B 於比賽的第 $26, 36, 46, 56$ 分鐘解出這些題目，隊伍 C 於比賽的第 $15, 25, 35, 45$ 分鐘解出這些題目。隊伍 A 分別在這些題目嘗試了 $1, 1, 1, 2$ 次後通過，隊伍 B 分別在這些題目嘗試了 $2, 3, 1, 4$ 次後通過，隊伍 C 分別在這些題目嘗試了 $1, 3, 1, 2$ 次後通過。

隊伍 A 的罰時為 $160 + 20 = 180$，隊伍 B 的罰時為 $164 + 120 = 284$，隊伍 C 的罰時為 $120 + 60 = 180$ 分鐘，因此隊伍 B 的名次會在另外兩支隊伍後面；而隊伍 A 最後一次答對題目的時間為 55 分鐘，隊伍 C 則為 45 分鐘，故隊伍 C 的排名又會在隊伍 A 之前。

為了讓隊伍了解即時的賽況，主辦單位會公布計分板顯示各隊的解題情形。計分板為一個 $N \times M$ 的表格，由上到下第 i 行表示第 i 支隊伍，由左到右第 j 列表示第 j 道題目。若第 i 支

隊伍已經解出了第 j 道題目，則計分板上該格會顯示「$s_{i,j}$/$m_{i,j}$」（不含引號），若第 i 支隊伍還沒解出第 j 道題目，則該格會顯示「$s_{i,j}$/--」（不含引號），其中 $s_{i,j}$ 表示其嘗試的次數，$m_{i,j}$ 表示該隊在比賽開始後 $m_{i,j}$ 分鐘解出該題。

比賽最刺激的莫過於有黑馬隊伍逆轉奪下第一了！現在比賽正準備進入第 T 分鐘，你想寫一支程式幫每一支隊伍算一算，如果其他隊伍都沒有再解出題目，他們至少要再解出幾道題目才有機會獨佔第一名的位置。也就是說，沒有隊伍的排名在他們前面，也沒有隊伍與他們排名並列。

Input

輸入的一行包含三個整數 N, M, T，分別以一個空白隔開，分別表示參賽隊伍數量、題目數量、目前比賽經過的時間。

接下來 N 行，每行包含 M 個字串，字串間分別以一個空白隔開。第 i 行的第 j 個字串為 $t_{i,j}$，表示第 i 支隊伍在第 j 道題目的解題狀況。若第 i 支隊伍已經解出了第 j 道題目，則 $t_{i,j}$ 為「$s_{i,j}$/$m_{i,j}$」（不含引號）；若第 i 支隊伍還沒解出第 j 道題目，則 $t_{i,j}$ 為「$s_{i,j}$/--」（不含引號）。$s_{i,j}$ 及 $m_{i,j}$ 皆為整數，其中 $s_{i,j}$ 表示其嘗試的次數，$m_{i,j}$ 表示解出該題的時間。

- $2 \leq N \leq 500$
- $1 \leq M \leq 50$
- $1 \leq T < 300$
- $0 \leq s_{i,j} \leq 100$
- $0 \leq m_{i,j} < T$
- 若隊伍 i 已經解出了第 j 道題目，則 $s_{i,j} > 0$

Output

輸出 N 行。對於第 i 行輸出，若第 i 支隊伍還有機會獨佔第一名的位置，請輸出一個整數，表示他們至少要再解出幾道題目才有機會獨佔第一名的位置；否則，請輸出「-1」（不含引號）。

Sample Input 1

```
3 3 1
0/-- 1/-- 2/--
7/-- 2/-- 7/--
3/-- 1/-- 4/--
```

Sample Output 1

```
1
1
1
```

Sample Input 2

```
5 5 60
1/25 1/35 1/45 8/-- 2/55
2/26 3/36 0/-- 1/46 4/56
2/-- 1/35 2/45 1/15 3/25
1/25 1/21 2/14 2/-- 3/--
1/24 1/21 2/14 2/-- 3/--
```

Sample Output 2

```
1
1
0
2
1
```

Sample Input 3

```
3 1 10
2/1
1/--
0/--
```

Sample Output 3

```
0
-1
1
```

Sample Input 4

```
7 13 240
1/8 1/33 1/21 1/46 1/113 1/38 0/-- 1/28 0/-- 1/61 0/-- 1/77 1/4
2/11 1/20 1/68 1/45 1/128 1/32 0/-- 1/25 0/-- 1/103 0/-- 1/78 1/3
1/11 1/20 1/39 1/30 1/107 1/17 0/-- 1/5 1/-- 1/85 0/-- 1/59 1/7
2/8 1/20 1/53 1/34 1/98 1/28 0/-- 1/12 4/-- 1/59 0/-- 1/73 1/14
1/5 1/22 1/60 1/44 1/89 1/50 0/-- 1/25 6/-- 1/79 0/-- 1/39 1/11
2/9 1/13 1/44 2/34 2/138 1/22 0/-- 1/16 1/-- 1/76 0/-- 1/93 1/3
1/12 1/15 1/40 1/32 13/-- 1/26 0/-- 1/19 0/-- 1/79 0/-- 1/57 1/4
```

Sample Output 4

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

B. 密碼測試

Problem ID: password

喔不，你的好朋友小良發現她最近的密碼沒有通過 NPCS（Noxious Password Checking System）測試，需要立刻更換一個新的密碼！

已知她現在的密碼是一個由小寫英文字母組成的字串，而要通過 NPCS 的測驗，你只要讓密碼中相鄰的字元都不相同就可以了。

然而現在這個密碼小良已經使用很久，對它有了感情，她不想要更改太多，因此她想要在裡面插入最少數量的底線字元（_），讓密碼可以通過 NPCS 測試。

然而小良不擅長處理這個問題，給你她的密碼，你能夠幫助她完成這件事嗎？

Input

輸入有一行，上面有一個由小寫英文字母組成的字串 s，代表小良的密碼。

- s 的長度介於 1 到 2000 之間

Output

請輸出一行，代表修改後的密碼，修改後的密碼需為通過 NPCS 且插入最少底線字元（_）的密碼。若有多個合法的解，你可以輸出任意一個。

Sample Input 1	Sample Output 1
abaacc	aba_ac_c

Sample Input 2	Sample Output 2
abcabcabc	abcabcabc

Sample Input 3	Sample Output 3
oooo	o_o_o_o

Sample Input 4	Sample Output 4
thissystemmakessense	this_system_makes_sense

Note

對於範例測試資料 1，以下為一些不符合要求的輸出例子：

- aba_acc：這個字串沒有通過 NPCS。
- ab_a_ac_c：雖然它會通過 NPCS，但它的長度不是最短的。
- abagachc：這個字串插入了 _ 以外的字元。

另外，雖然小良告訴你她的密碼，請不要盜用她的帳號喔！

C. 巫醫巫醫滅滅滅

Problem ID: light

巫醫巫醫最擅長出滅台題了，所謂滅台題指的就是在一場比賽中沒有任何人得到分數的題目。今年滅台題多到湊出了一場巫醫巫醫滅台邀請賽。

雖然所有的參賽者在報名之前都知道比賽的結果，報名仍然相當踴躍，因此巫醫巫醫不得不多租借幾個場地以容納來自世界各地的粉絲，其中一個場地可以被劃分為 $N \times M$ 的表格，橫列編號為 0 至 $N-1$、直排編號為 0 至 $M-1$。

每個格子都有一盞燈，但可能因為場地狀況、參賽者放棄參加等等的因素，所以不見得每個格子都有參賽者。我們用 $a_{i,j}$ 代表這個格子的狀態，如果 $a_{i,j}$ 為 P 表示第 i 列第 j 行有參賽者，反之是 . 則沒有參賽者。

正是因為如此，巫醫巫醫發現場地有些地方可以不需要照亮！不過要是一個一個格子都調整燈光實在是太費時又不美觀，為此她決定選擇只開一些橫列以及一些直行的燈，正式的說，她會用以下的方式打開會場的燈：

1. 一開始所有燈都是關閉的。
2. 決定四個整數 l, r, u, d，滿足 $0 \le u \le d \le N$ 且 $0 \le l \le r \le M$。
3. 巫醫巫醫會依序打開 $u, u+1, u+2, \ldots, d-1$ 橫列的所有燈，總共花費 $d-u$ 單位的時間，注意第 d 列燈的狀況**沒有**被更動。
4. 巫醫巫醫會依序打開 $l, l+1, l+2, \ldots, r-1$ 直排的所有燈，總共花費 $r-l$ 單位的時間，注意第 r 排燈的狀況**沒有**被更動。

注意如果 $u = d$，那巫醫巫醫在第三步中不會做任何事。同樣如果 $l = r$，那巫醫巫醫在第四步中不會做任何事。

巫醫巫醫想要讓所有參賽者所在的格子，燈光都是開啟的，不過巫醫巫醫正忙著把眾多滅台題上傳至系統中，請你幫她寫一個程式算出最少的燈光設置時間。

Input

輸入第一行有兩個以空白分開的整數 N, M，接下來有 N 行，每行有一個長度為 M 的字串，第 i 行的第 j 個字元代表 $a_{i-1,j-1}$。

- $1 \leq N, M \leq 5000$
- $a_{i,j}$ 是 P 或 .

Output

輸出一行，表示最少的燈光設置時間，也就是滿足條件的 $(d-u)+(r-l)$ 最小值。

Sample Input 1	Sample Output 1
5 5 ....P ..... PP..P ...P. ...P.	3

Note

在範例中，巫醫巫醫可以這樣花費最少的時間打開滿足條件的燈光：

- 選擇 $u = 2, d = 3$。只有第 2 橫列的燈會被打開，花費 1 單位的時間。
- 選擇 $l = 3, r = 5$。第 $3, 4$ 直排的燈會被打開，花費 2 單位的時間。

D. 李杈搭電梯

Problem ID: elevator

李杈去參加國際大學生西洋棋競賽，作為一個頂級的世界級比賽，主辦單位為每一位參賽選手準備了一間八星級飯店中的豪華單人套房，讓選手在辛苦的賽程之餘，也能好好地休息。

不幸的是，李杈很怕搭電梯，而這間飯店的樓梯間沒什麼人走，看起來很陰森，因此李杈不得不搭乘電梯上下樓。這間飯店地面上有九層樓，地面下也有九層樓，總共有 18 層樓，地上一樓是大廳，這層樓被記為 L，而地上二樓到地上九樓分別是 2、3、……、9，地下一樓到地下九樓分別是 B1、B2、……、B9。

現在，李杈又要搭電梯了。走進電梯時，他看到目前顯示的樓層是樓層 x，並且他按下了樓層 y，他想知道從樓層 x 到樓層 y 要移動幾層樓，好知道他要花多少時間才能離開這個可怕的地方。

Input

第一行有一個整數 T，代表有幾筆測試資料。

接下來有 T 行，每行包含兩個字串 x, y，代表李杈想要從樓層 x 到樓層 y。

- $1 \leq T \leq 500$
- x, y 只會是 L,2,...,9,B1,B2,...,B9
- $x \neq y$

Output

輸出 T 行，其中第 i 行輸出一個整數，代表第 i 筆測試資料中，李杈從樓層 x 到樓層 y 要移動幾層樓。

Sample Input 1	Sample Output 1
6 3 6 L 5 B3 B5 9 B9 4 L B1 L	3 4 2 17 3 1

E. 果樹大盜

Problem ID: trees

王董有一塊地，這塊地的大小為 $N \times M$，為了方便，他將這塊地切成同寬度的 N 個橫列和 M 直行，並讓 (x, y) 代表由上數來第 x 列由左數來第 y 行所切出的 1×1 的小地。

對於所有 $N \times M$ 個大小 1×1 的小地，上面最多種了一棵水果樹，所種的水果一定是芒果或奇異果其中一種。方塊王想要趁王董出差的時候，偷走所有的芒果樹和奇異果樹，但是王董有設置特殊的警報系統，所以這個任務並不簡單。

經過一番調查，方塊王發現一次行動只能偷芒果樹和奇異果樹各一棵，而且它們之間不能有其他芒果樹或奇異果樹，否則絕對會觸發警報。更明確的說，一次行動中，方塊王只能選擇一塊種有芒果樹的小地和一塊種有奇異果樹的小地，而且以這兩塊小地為角落所形成的長方形區域內不能有其他的果樹。注意到一次行動後，被偷的小地就不再種有芒果樹或奇異果樹。

在以上三張圖中，芒果樹和奇異果樹分別用芒果和奇異果表示。在左邊和中間的圖中，方塊王可以拿走用藍色圓圈圈起來的兩棵果樹，因為兩棵果樹為角落所形成的長方形區域（紅色框框）中沒有其他的果樹。在右圖中，方塊王不能拿走圈起來的兩棵果樹，因為它們為角落所形成的長方形區域內還有別的果樹。

方塊王是個完美主義者，所以如果沒辦法把全部的樹都偷走，那他就會直接放棄偷樹，因此他想知道是否有辦法透過數次行動將所有的樹都偷走？

請幫方塊王找到任何透過數次行動將所有的樹都偷走的方案，或是向他說明這不可能辦到。

Input

輸入的第一行有兩個整數 N, M，代表王董地的大小。

接下來會輸入 N 行，第 i 行輸入 M 個字元 $c_{i,1}, c_{i,2}, \ldots, c_{i,M}$，其中若 $c_{i,j} =$ `.`，則代表 (i, j) 沒種任何水果樹，$c_{i,j} =$ `o` 代表種了一棵芒果樹，而 $c_{i,j} =$ `x` 則代表種了一棵奇異果樹。

- $1 \leq N, M \leq 1000$
- 對於所有 $1 \leq i \leq N, 1 \leq j, \leq M$，$c_{i,j}$ 一定是 `.`, `o`, `x` 其中一個。

Output

如果不存在把全部的樹都偷走的方法，則輸出「`-1`」（不含引號）即可。

否則，第一行輸出一個整數 K，代表要進行 K 次行動。

接下來輸出 K 行，其中第 i 行輸出四個整數 x_1, y_1, x_2, y_2，代表這次行動偷了位於 (x_1, y_1) 和 (x_2, y_2) 的芒果樹和奇異果樹，這兩塊小地哪個是芒果樹，哪個是奇異果樹並不重要，只要其中一個是芒果樹，另一個是奇異果樹即可。

Sample Input 1

```
4 4
.o.x
x.oo
.xo.
x...
```

Sample Output 1

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

Sample Input 2

```
3 3
oxo
xox
oxo
```

Sample Output 2

```
-1
```

Sample Input 3	Sample Output 3
5 5 o.o.o .x.x. .x.x. .x.x. o.o.o	6 1 5 2 4 1 3 2 2 3 2 1 1 5 5 4 4 5 3 4 2 3 4 5 1

Sample Input 4	Sample Output 4
5 3 oxo o.x xxx ..o ..o	5 1 1 1 2 2 1 2 3 1 3 3 3 4 3 3 2 5 3 3 1

This page is intentionally left blank.

F. 烏龜疊疊疊

Problem ID: turtles

烏龜們喜歡像烏龜一樣疊在一起。現在，有 N 隻烏龜排成一橫排，編號由左至右為 1 到 N ，編號 i 的烏龜體重是 w_i。他們會依照以下步驟疊在一起：

1. 一開始，每一隻烏龜自成一堆，總共有 N 堆。
2. 每一輪，原本第一堆的烏龜會整堆爬到原本第二堆的烏龜上面形成新的一堆，原本第三堆的烏龜會整堆爬到第四堆的烏龜上面，依此類推。如果這輪開始前總堆數是奇數的話，那最後一堆會留在原地不動。
3. 如果還有超過一堆的烏龜，會再進行下一輪，直到只剩下一堆烏龜為止。

然而，背著其他烏龜是很辛苦的。為了了解每隻烏龜的辛苦程度，請你求出最後每一隻烏龜所要背的烏龜的體重總和吧！

Input

輸入有兩行。第一行只有一個整數 N，代表有 N 隻烏龜。第二行有 N 個整數 $w_1, w_2, \cdots, w_N$，第 i 個數代表第 i 隻烏龜的體重。

- $1 \leq N \leq 1000$
- $1 \leq w_i \leq 10^6$

Output

輸出一行包含 N 個整數，第 i 個數代表編號 i 的烏龜在疊完後所要背的烏龜的體重總和。

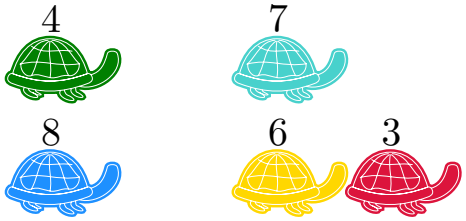
Sample Input 1	Sample Output 1
5 4 8 7 6 3	0 4 12 19 25

Note

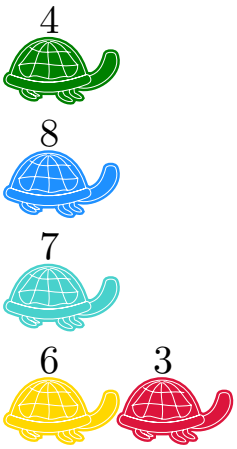
範例測試資料的過程如下，每隻烏龜上的數字代表他的體重：

一開始的排列：

第一輪完的排列：



第二輪完的排列：



第三輪完的排列，也是最後的排列：

This page is intentionally left blank.