2023 網際網路程式設計全國大賽
國中組決賽

- 本次比賽共 7 題，含本封面共 22 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	月亮	1 s	1024 MB
B	板子	1.5 s	1024 MB
C	喵喵星球	1 s	1024 MB
D	旅行	1 s	1024 MB
E	猜三個數字	1 s	1024 MB
F	召喚大神	1 s	1024 MB
G	方城之戰	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. 月亮

Problem ID: moon

眾所周知，月亮是地球的一個天然衛星，自古以來有許多傳說都與月亮有關。

月球本身不發光，月球直接被太陽照射之部份反射太陽光，才可見發亮，其陰影部分是月球自己之陰暗面。根據天文學，月球環繞地球公轉時，地球、月球、太陽之相對位置不斷規律地變化，使觀測者從不同角度看到月球被太陽照明之部分，造成月相盈虧圓缺之變化。

當地球位於月球和太陽之間時，我們可以看到整個被太陽直射的月球部分，這就是滿月。當月球位於地球和太陽之間時，我們只能看到月球不被太陽照射的部分，這就是新月。當地球與月球的連線和太陽與月亮的連線正好成直角時，我們正好可以看到月球被太陽直射的部分的一半，這就是弦月。

以上的敘述參考自維基百科。

對宇宙很感興趣的小幽和海果，她們拿起了月亮模型來模擬月亮的盈虧圓缺。小幽和海果在桌子上做實驗，因為桌子很平，你可以假設桌上的東西的位置都可以用兩個數字 (x, y) 的直角座標系描述。小幽把光源設置在位置 (x_1, y_1)，照射的方向是從光源放置的位置往月亮模型的中心照射。海果在位置 (x_2, y_2) 作為觀察者觀察。請問海果看到的月相是滿月、新月、弦月，還是以上皆非？你總共會需要回答 T 個這樣的問題。

月亮模型為一個球心位在**原點**的球體，光源可以被視為是平行光源，並且海果在觀察時不會擋住光源造成影子。

因為小幽和海果是用月亮模型模擬月亮盈虧圓缺，光源有可能會位在觀察者和月亮模型之間。在光源恰好落在觀察者和月亮模型所形成的線段上時，請當成是滿月的情況。

Input

輸入的第一行包含一個整數 T，表示總共你被問了幾次問題。

接下來 T 行，每行包含四個以空白隔開的整數 a, b, c, d，表示光源在 (a, b)，而觀察者在 (c, d)。

- $1 \leq T \leq 10^5$

- $-10^9 \leq a, b, c, d \leq 10^9$
- 保證光源和觀察者都不在原點，即「$a \neq 0$ 或 $b \neq 0$」且「$c \neq 0$ 或 $d \neq 0$」。

Output

輸出 T 行，表示問題的答案。對於第 i 個問題，請輸出恰一行字串。如果答案是滿月，請輸出「full」；如果答案是新月，請輸出「new」；如果答案是弦月，請輸出「half」；否則請輸出「neither」。以上所述均不含引號。

Sample Input 1

```
7
-1 -9 1 9
6 1 1 -6
2 2 6 6
3 6 -2 -4
-6 -2 -1 3
-6 -7 6 7
10 4 10 4
```

Sample Output 1

```
new
half
full
new
half
new
full
```

Sample Input 2

```
10
-960577638 -621915527 621915527 -960577638
-908624183 371306062 -908624183 371306062
-173999741 -863137243 -863137243 173999742
-811414735 615988003 -811414735 615988003
-920394483 -989725511 -920394483 -989725511
-449755957 503148141 449755957 -503148141
293886964 929048866 -293886965 -929048865
-385725563 653935775 653935775 385725563
488174711 -827297097 488174709 -827297096
-363146704 -375940411 726293408 751880822
```

Sample Output 2

```
half
full
neither
full
full
new
neither
half
neither
new
```

B. 板子

Problem ID: board

小波是個板子收藏家，他家裡有很多各式各樣的板子，像是有個板子上面寫著如何求出三維凸包、有個板子上面記載著如何以 $O(E \log V)$ 的時間複雜度找出有向最小生成樹。也有的板子充滿著神奇的魔法。

有一天，他獲得了一個新的板子，上面畫有 $N \times N$ 的棋盤，N 個橫列由上至下從 1 到 N 編號、N 個直行由左至右從 1 到 N 編號，第 i 橫列第 j 直行的格子記作 (i, j)。

這個板子具有神奇的魔力，如果在一顆石頭上寫下 U、D、L、R 的其中一個字母，並把它放到棋盤的一個格子上，這個石頭每過一秒就會自己根據它上面所寫的字母，往當前格子的上、下、左或右（分別對應到 U、D、L、R）移動一格。如果會移動到超出邊界的話，那麼這顆石頭會直接穿越空間到棋盤的對面，也就是說：

- 對於 $1 \le r \le N$，如果有一顆石頭要從 $(r, 1)$ 往左移動一格，那麼它會移動到 (r, N)。
- 對於 $1 \le r \le N$，如果有一顆石頭要從 (r, N) 往右移動一格，那麼它會移動到 $(r, 1)$。
- 對於 $1 \le c \le N$，如果有一顆石頭要從 $(1, c)$ 往上移動一格，那麼它會移動到 (N, c)。
- 對於 $1 \le c \le N$，如果有一顆石頭要從 (N, c) 往下移動一格，那麼它會移動到 $(1, c)$。

小波覺得這個板子很好玩，然而，要是有兩顆石頭相撞的話，板子上的石頭就會全部飛出去，小波還得花很多時間把石頭們撿回來，太累了。因此，小波必須在放下石頭之前仔細的思考有沒有可能會有兩顆石頭相撞。兩顆石頭相撞的條件是它們在同一個時間點移動到同一個格子，或者它們在同一個時間點要移動到對方本來所在的位置。正式地說，如果有兩顆石頭本來分別在 $(r_1, c_1), (r_2, c_2)$，下一秒分別會移動到 $(r'_1, c'_1), (r'_2, c'_2)$，如果滿足以下兩個條件其中之一，它們就會相撞：

- $(r'_1, c'_1) = (r'_2, c'_2)$
- $(r_1, c_1) = (r'_2, c'_2)$ 而且 $(r_2, c_2) = (r'_1, c'_1)$

小波準備了 K 顆石頭，並且已經在上面寫好了代表它們各自移動方向的字母，第 i 顆石頭上面寫的字母是 d_i。他想要將第 i 顆石頭放在格子 (r_i, c_i)，他不會想要把兩顆石頭放在同一個格子，那是在自找麻煩。請告訴他在放下所有石頭之後（他可以用某個板子上記載的魔法同時把所有石頭放到他想要的位置），把板子一直放著的話，會不會有石頭在未來的某個時刻相撞。

Input

第一行有一個整數 T，代表有幾筆測試資料。

每一筆測試資料的第一行有兩個整數 N, K，分別代表板子上棋盤的大小與小波準備的石頭數量。

接下來有 K 行，其中第 i 行有一個字母和兩個整數 d_i, r_i, c_i，代表第 i 顆石頭上面寫的字母是 d_i，小波想把它放在 (r_i, c_i)。

- $1 \leq T \leq 10^6$
- $3 \leq N \leq 10^6$
- $1 \leq K \leq 10^6$
- $K \leq N^2$
- 在全部 T 筆測試資料中，K 的總和 $\leq 10^6$
- 在全部 T 筆測試資料中，N 的總和 $\leq 10^7$
- 對於 $1 \leq i \leq K$，$1 \leq r_i, c_i \leq N$
- 同一筆測試資料中，對於 $i \neq j$，$(r_i, c_i) \neq (r_j, c_j)$
- 對於 $1 \leq i \leq K$，d_i 是 U、D、L、R 的其中之一

Output

對於每一筆測試資料，如果永遠不會有兩顆石頭相撞，輸出「Good」，否則，輸出「Bad」(不含引號)。

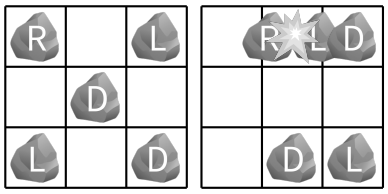
Sample Input 1	Sample Output 1
5 5 2 L 3 1 R 3 5 3 5 R 1 1 L 1 3 D 2 2 L 3 1 D 3 3 3 5 R 1 1 R 1 3 D 2 2 L 3 1 D 3 3 4 4 U 2 1 U 2 3 R 3 3 U 4 1 4 7 D 4 4 D 2 4 U 3 3 U 4 2 U 4 3 U 1 3 D 1 1	Bad Bad Bad Good Good

Note

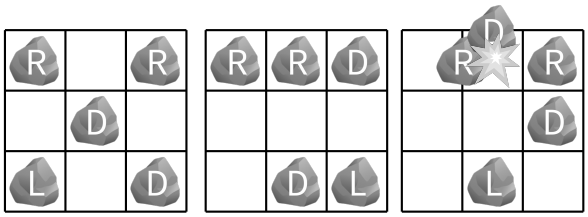
在範例測試資料的第一筆測試資料中，兩顆石頭一開始分別在位置 $(3,1)$ 和 $(3,5)$，移動方向分別是左和右，經過一秒後它們會分別移動到 $(3,5)$ 和 $(3,1)$，由於它們交換了位置，所以他們相撞了。

第二筆測試資料中，第 1 顆和第 2 顆石頭會在經過一秒後試圖移動到同一個格子並且相

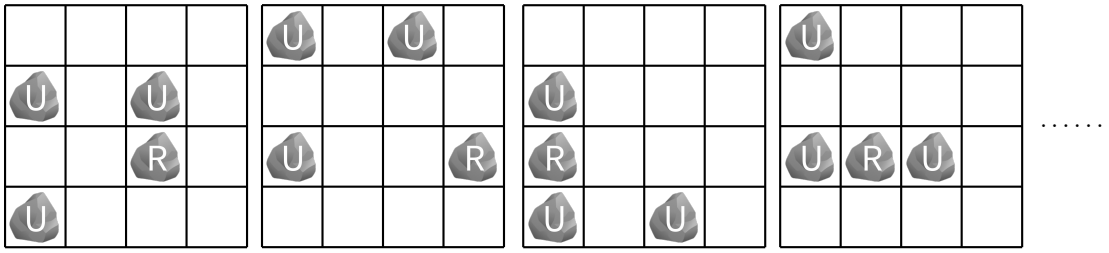
撞。以下左圖是一開始石頭的位置，右圖是經過一秒後石頭的位置，爆炸圖案代表兩顆石頭在那裡相撞。



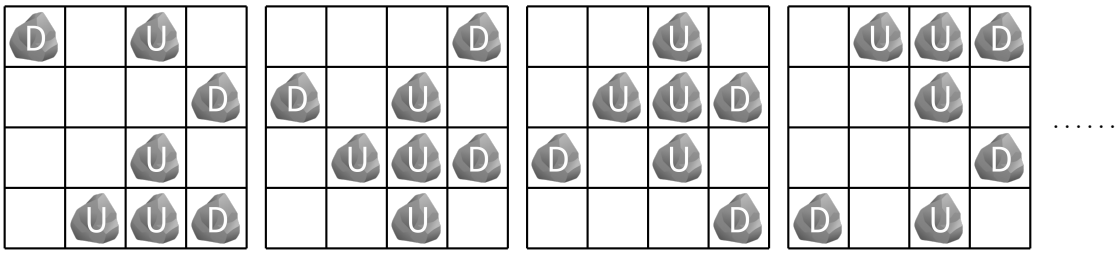
以下是第三筆測試資料中，從一開始到發生相撞時石頭的移動過程。



下圖是第四筆測試資料中，從開始到三秒後的石頭移動過程。無論過了多久，永遠都不會有石頭相撞。



下圖是第五筆測試資料中，從開始到三秒後的石頭移動過程。



C. 喵喵星球

Problem ID: meowmeow

喵喵，喵喵喵，喵喵喵喵，喵喵喵喵喵？

喵喵星球是一個太陽系以外的星球，在那裡一共有 26 種貨幣，價值低到高依序為 A, B, C 一直到 Z。其中 1000 個 A 貨幣的價值與 1 個 B 貨幣相同，1000 個 B 貨幣的價值與 1 個 C 貨幣相同，以此類推。

居住在喵喵星球的媚媚現在有 x_1 個 y_1 貨幣，她想知道她能不能購買一個需要 x_2 個 y_2 貨幣才買得起的東西呢？請寫一個程式幫助她判斷。

喵喵喵喵喵，喵喵喵喵，喵喵喵，喵喵！

Input

輸入有兩行，第一行有以空格隔開的一正整數與一大寫英文字母 x_1, y_1，代表媚媚現在有 x_1 個 y_1 貨幣。

第二行有以空格隔開的一正整數與一大寫英文字母 x_2, y_2，代表媚媚想要買的東西需要 x_2 個 y_2 貨幣。

- $1 \leq x_1, x_2 < 10^{100}$
- x_1, x_2 皆沒有前導 0。
- y_1, y_2 為一大寫英文字母。

Output

若媚媚可以購買她想要買的東西，請輸出 Yes，否則請輸出 No。

Sample Input 1	Sample Output 1
345 B 123456 A	Yes

Sample Input 2	Sample Output 2
345 B 345678 A	No

Sample Input 3	Sample Output 3
1000000000 G 1000 I	Yes

Sample Input 4	Sample Output 4
864197532123456789 O 864197532123456798 O	No

Sample Input 5	Sample Output 5
3141592653589793238462643383327 A 1 Z	No

Note

請特別注意輸入可能不能用 64-bit 整數存下。

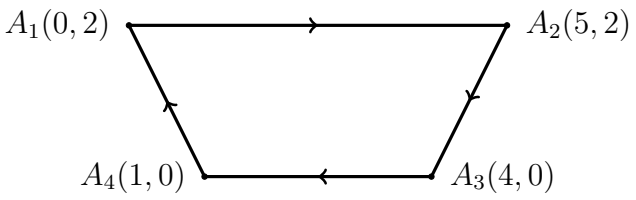
D. 旅行

Problem ID: travel

經過了一個學期的努力終於到假期了！小桃決定用一趟旅行來慶祝假期的開始。今天小桃拿出了一張地圖，並在上面標上一些她想要去的景點。地圖為一個二維平面，而每個景點都會是二維平面上的一個整數點。已知小桃一共想要去 N 個景點，且第 i 個景點在座標 (x_i, y_i) 上，任兩個景點的座標都不相同。

小桃有著獨特的方法來規劃她的旅行路線，首先她會先選擇 M 個相異的整數點 $A_1, A_2, \ldots, A_M$，接著她會從 A_1 出發，以最短距離移動到 A_2，再以最短距離移動到 A_3，以此類推。而當移動到 A_M 後再以最短距離回到 A_1。

舉例來說，若 $M = 4, A_1 = (0, 2), A_2 = (5, 2), A_3 = (4, 0), A_4 = (1, 0)$，那她移動的路線會是：



小桃希望旅行路線能夠經過所有的景點，同時她不希望旅行的路線經過重複的點，意即除了起終點相同外，路線不會相交。此外，小桃希望挑選的點數量至少為 3 且至多為 2000。現在給你全部的景點，你能幫幫小桃找到一條符合所有條件的路線嗎？

Input

輸入第一行有一個正整數 N，表示小桃想要去的景點數量接下來 N 行，第 i 行有兩個用空格隔開的整數 x_i, y_i，代表第 i 個景點的 x 座標與 y 座標。

- $3 \leq N \leq 1000$
- $-10^6 \leq x_i, y_i \leq 10^6$
- $\forall i \neq j, (x_i, y_i) \neq (x_j, y_j)$

Output

第一行請輸出一個正整數 M，表示你要挑選的整數點。

接下來 M 行，第 i 行請輸出兩個用空格隔開的整數 a_i, b_i，代表第 i 個挑選的點的 x 座標與 y 座標。

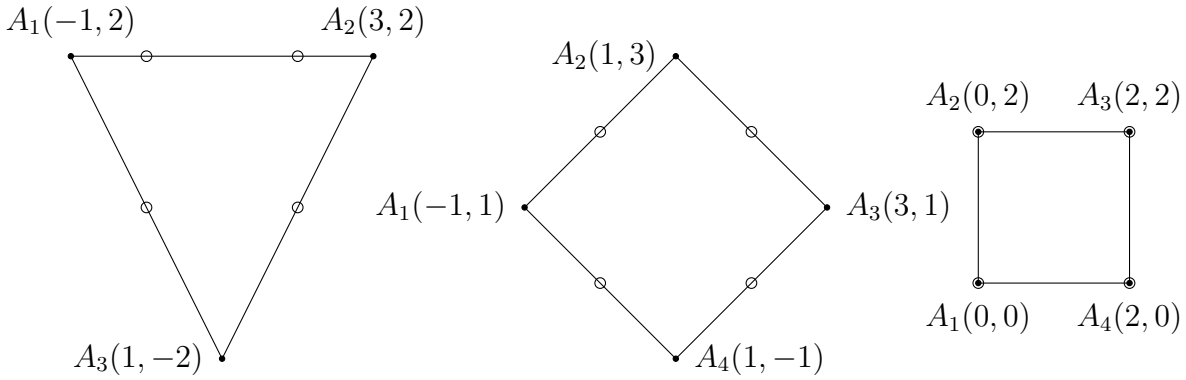
你的答案會被視為正確若你符合以下條件：

- $3 \leq M \leq 2000$
- $-10^9 \leq a_i, b_i \leq 10^9$
- $\forall i \neq j, (a_i, b_i) \neq (a_j, b_j)$
- 旅行的路線會經過所有景點
- 旅行的路線不會經過重複的點

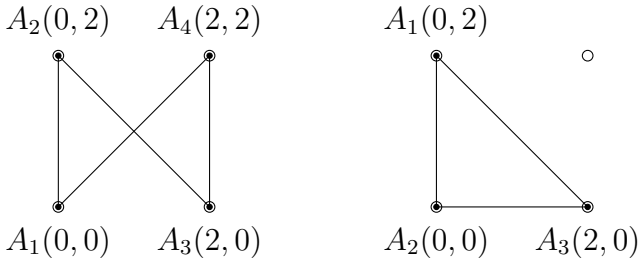
Sample Input 1	Sample Output 1
4 0 0 0 2 2 0 2 2	3 -1 2 3 2 1 -2

Note

範例測試資料一中，以下為一些合法的旅行路線，這裡空心的圓點代表景點。



以下為一些不合法的旅行路線，左邊的因為路線經過了點 $(1,1)$ 兩次，右邊的因為她沒有經過點 $(2,2)$。



你可以運用以下方格紙來幫助你思考：

This page is intentionally left blank.

E. 猜三個數字

Problem ID: 3black

本題為互動題。

我在心中想了三個不同的整數，他們都介於 1 到 200，你有辦法猜到這三個數字是多少嗎？每當你猜了一組數字，我可以告訴我想的數字有奇/偶數個在你猜的數字裡（注意到 0 是偶數）。但你最多只能猜 33 次，所以你要好好選擇你猜的數字。

互動說明

當你的程式打算要猜數字時，輸出一行且包含一個整數 K，並在下一行輸出 K 個兩兩相異的整數，這些整數必須介於 1 到 200 之間（包含 $1, 200$）。當你猜完數字後，記得要清空（flush）標準輸出（standard out）。

當我們收到你的猜測後，會把你猜的結果回覆到你的標準輸入（standard in）。回覆會是下列兩種：

- 『odd』如果你猜的 K 個整數中有奇數個數字和我心中想的數字一樣
- 『even』如果你猜的 K 個整數中有偶數個數字和我心中想的數字一樣
- 『correct』如果 $K = 3$ 而且你猜的數字都和我心中想的數字一樣

當你猜到了正確數字後，你的程式必須立刻結束（exit）。我會回答至多你 33 次，如果這 33 次都沒有回覆你 correct（，你的程式將會被強制中止。注意到我在過程中可能會改變心意，換一組數字，不過我保證就算換了一組數字，之前的回答依舊符合新的答案。

以下是 C 程式 flush 的範例：

```c
#include <stdio.h>

int main()
{
  printf( "500\n" );
  fflush( stdout );
}
```

以下是 C++ 程式 flush 的範例：

```cpp
#include <iostream>
int main()
{
  std::cout << "500\n";
  std::cout << std::flush;
}
```

Sample Input	Sample Output
odd even correct	5 1 2 3 4 5 4 1 2 4 5 3 2 4 3

F. 召喚大神

Problem ID: summon

小魚在做一道極其困難的題目，他已經做了好幾個禮拜，但仍然做不出來。

「這題太難了啦，真希望有大神能助我一臂之力。」小魚自言自語地說道。

剛好路過的精靈小波聽到小魚的哀號聲，決定送給他一個神奇的道具。「試試看這個吧！」精靈小波邊說邊遞給小魚一支筆。

「這支筆可不是一般的筆，只要用這支筆在地上畫出一個合法的魔法陣，就能召喚出大神來。」精靈小波解釋道。

「真的假的，謝謝你精靈小波。」小魚像是終於從枷鎖中解脫般不斷向精靈小波道謝。

一個合法的魔法陣是由 $N \times N$ 個小寫英文字母排成一個 $N \times N$ 的正方形所形成的圖案，為了保證魔法陣能正常的啟動，還有幾個特殊的條件：

- 四個角落的英文字母必須是相同的。
- 在四個邊上且不是角落的 $4N - 8$ 個英文字母必須是相同的。
- 剩下的所有英文字母都必須是相同的。

小魚患有選擇障礙症，他知道能正常啟動的合法魔法陣有很多種，但他不知道該畫哪一種比較好。

請幫小魚畫出任意一個能正常啟動的合法魔法陣。

Input

輸入只有一個正整數 N，表示魔法陣的大小為 $N \times N$。

- $3 \leq N \leq 100$

Output

請輸出 N 行，每一行輸出 N 個小寫英文字母 $c_1c_2\cdots c_N$。第 i 行輸出的第 j 個小寫英文字母代表魔法陣從上面數來第 i 列、從左邊數來第 j 行的小寫英文字母。

如果有多種能正常啟動的合法魔法陣，輸出任意一種即可。

Sample Input 1	Sample Output 1
5	zcccz ckkkc ckkkc ckkkc zcccz

G. 方城之戰

Problem ID: square

在過去的 NPSC，你已經經歷過了國士無雙、七對子等日本麻將相關的題目了。你相信你現在是一位日本麻將的專家，任何日本麻將相關的題目都難不倒你！

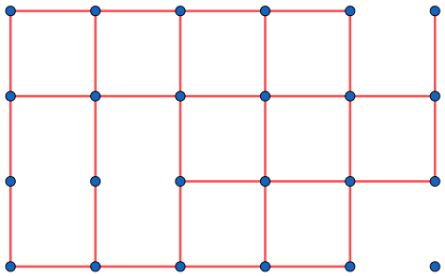
你說的對，但是「方城之戰」是由「爾伊栖」自主研發的一款全新開放世界冒險遊戲。遊戲發生在一個被稱作「沃塞遜」的幻想世界，在這裡，被神選中的人將被授予「蘇杜爾」，導引運算之力。你將扮演一位名為「小方塊」的神秘角色，在電子之海的旅行中邂逅性格各異、能力獨特的同伴們，和他們一起突破重圍，找回失散的碎片——同時，逐步發掘「方城」的真相……

在「方城之戰」中，一座大小為 $N \times M$ 的城市由 $(N+1) \times (M+1)$ 座瞭望臺構成。所有的瞭望臺排列成 $N+1$ 個橫行及 $M+1$ 個直列，其中每一行由上到下以 0 到 N 的整數編號，每一列由左到右依序以 0 到 M 的整數編號，第 i 行第 j 列的瞭望臺座標為 (i, j)。城市中同一行或同一列相鄰的兩座瞭望臺之間距離皆為 1 公里，並且相鄰的兩座瞭望臺之間可能蓋有一座城牆。

為了突破這些城牆的封鎖，小方塊需要先分析這座城市的強度。小方塊定義一座城市的強度為這些城牆所組成的「方城」的數量。一座「方城」的四個頂點須由相異的四座瞭望臺 $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$ 組成，並且這些瞭望臺必須滿足以下的所有條件：

- $x_1 = x_2 < x_3 = x_4$
- $y_1 = y_4 < y_2 = y_3$
- $x_3 - x_1 = y_3 - y_1$
- 對於所有的 $x_1 \le a < x_3$，瞭望臺 (a, y_1) 與 $(a+1, y_1)$ 之間都蓋有一座城牆
- 對於所有的 $x_1 \le b < x_3$，瞭望臺 (b, y_3) 與 $(b+1, y_3)$ 之間都蓋有一座城牆
- 對於所有的 $y_1 \le c < y_3$，瞭望臺 (x_1, c) 與 $(x_1, c+1)$ 之間都蓋有一座城牆
- 對於所有的 $y_1 \le d < y_3$，瞭望臺 (x_3, d) 與 $(x_3, d+1)$ 之間也都蓋有一座城牆

以下圖為例，下圖為一座大小 3×5 的城市，其中圓點表示瞭望臺的位置，左上角的瞭望臺座標為 $(0, 0)$，兩個圓點之間的線段表示瞭望臺之間的城牆。$(1,0), (1,2), (3,2), (3,0)$ 四座瞭望臺組成一座方城，$(2,3), (2,4), (3,4), (3,3)$ 四座瞭望臺組成一座方城。而 $(0,0), (0,2), (2,2), (2,0)$、$(0,4), (0,5), (1,5), (1,4)$、$(1,2), (1,4), (2,4), (2,2)$ 等四座瞭望臺則沒有組成一座方城。



請你寫一支程式幫助小方塊計算一座城市中的方城數量，好讓小方塊早日攻入沃塞遜中，找回失散的碎片。

Input

輸入的第一行包含兩個整數 N, M，以一個空白隔開，表示城市的大小。

接下來 $2N + 1$ 行，每行包含一個長度為 $2M + 1$ 的字串，第 i 行的第 j 個字元意義如下：

- 若 i 及 j 皆為奇數，則該字元為「#」，表示座標為 $(\frac{i-1}{2}, \frac{j-1}{2})$ 的瞭望臺。
- 若 i 及 j 皆為偶數，則該字元為「.」，表示瞭望臺之間的空地。
- 若 i 為奇數且 j 為偶數，則該字元表示座標為 $(\frac{i-1}{2}, \frac{j}{2} - 1)$ 及 $(\frac{i-1}{2}, \frac{j}{2})$ 兩座瞭望臺之間是否有城牆。若該字元為「-」，則表示兩座瞭望臺之間有城牆；若該字元為「.」，則表示沒有。
- 若 i 為偶數且 j 為奇數，則該字元表示座標為 $(\frac{i}{2} - 1, \frac{j-1}{2})$ 及 $(\frac{i}{2}, \frac{j-1}{2})$ 兩座瞭望臺之間是否有城牆。若該字元為「|」，則表示兩座瞭望臺之間有城牆；若該字元為「.」，則表示沒有。

- $1 \leq N, M \leq 20$

Output

輸出一個整數，表示城市中的方城數量。

Sample Input 1

```
3 5
#-#-#-#-#.#
|.|.|.|.|.|
#-#-#-#-#-#
|.|.|.|.|.|
#.#.#-#-#-#
|.|.|.|.|..
#-#-#-#-#.#
```

Sample Output 1

```
15
```

Sample Input 2

```
3 3
#-#-#-#
|.|.|.|
#-#-#-#
|.|.|.|
#-#-#-#
|.|.|.|
#-#-#-#
```

Sample Output 2

```
14
```

Sample Input 3

```
1 1
#.#
...
#.#
```

Sample Output 3

```
0
```

This page is intentionally left blank.