少年圖靈計畫
Young Turing Program

0_送分題 – Hello World

(30分)

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~Z][a~z]、空格，以及常用英文符號。

資料範圍

[A~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!
```

1_股市小達人 (Stonks)

(10分)
時間限制: 1 second
記憶體限制: 256 MB

題目敘述

Nathan 最近在學習理財，在評估了各種理財方式的投資報酬率後，他決定開始接觸股票這項理財方式，因而進了股市，究竟他能不能獲利呢？

假設 Nathan 在股市花了 X 元購買了某間神奇公司的一張股票，接著，這間公司會經歷 N 個季度。每個季度，這間公司會將他們的股價先更新為 a_i ，然後他們會發放股息，持股者每擁有一張股票，就會收到 $\lfloor \frac{a_i}{100} \rfloor$ 元作為股息。Nathan 會將這些股息存起來，並在每個季度收到股息後用這些存下來的股息盡可能的購買更多該公司的股票，直到他所剩下的股息不足以購買股票為止。

在第N個季度，Nathan 會在收到股息後將所有股票賣掉，並檢查他有沒有獲利。你能夠幫他寫一個程式來快速檢查嗎？

輸入格式

第一行包含兩格正整數 N, X 代表季度的數量以及一開始買進股票的價格。

第二行包含 N 個正整數，第 i 個數字 a_i 代表第 i 個季度的股價。

輸出格式

請檢查 Nathan 在第 N 個季度賣掉所有股票後所賺的錢加上股息後是否比買進股票所花的錢還多，如果有，輸出 "stonks" (不包含引號)，否則輸出 "not stonks" (不包含引號)。

資料範圍

- $1 \leq N, X, a_i \leq 1000$

測試範例

輸入範例 1

```
5 100
45 55 51 100 120
```

輸出範例 1

```
stonks
```

輸入範例 2

```
5 100
120 140 80 90 70
```

輸出範例 2

```
not stonks
```

輸入範例 3

```
10 100
130 180 200 300 400 500 250 100 15 60
```

輸出範例 3

```
stonks
```

範例說明

在範例輸入1, 2中，Nathan 總共只有一張股票，並賺了 2 元股息。

在範例輸入3中，Nathan 在前 8 個季度賺了 19 元股息，並在第 9 個季度買了一張股票。最後一共擁有 2 張股票與 4 元股息。

2_一起學日語 (Learn_Japanese)

(10分)
時間限制: 1 second
記憶體限制: 256 MB

題目敘述

有一天，Nathan在寫作業的時候，看到雞塊的桌子上放著一封信，寫著以下內容：

> 尊敬なる雞塊様
>
> お世話になっております。恐れ入りますが、私どもより大切なお知らせがございます。YTP大会に関して、ご参加いただけることを心よりお願い申し上げます。
>
> 貴殿のお名前は、その創造性と技術によって多くの人々に尊敬されております。ですから、このような大会にご参加いただけることは、私たちにとって非常に光栄でございます。

不會日文的Nathan當然看不懂這封信。這時，雞塊剛好走過來解釋。原來，這是一封用日文所寫，邀請雞塊去參加ytp比賽的信件。看到雞塊的日文這麼好，Nathan好羨慕，於是，他決定利用閒暇時間來學習日文。

不過，Nathan很快就在學習動詞的變化時卡關了，日文動詞的辭書型轉ます型實在太難了！無助的他只好去向雞塊請教。雞塊告訴他，首先，日文動詞一定會以ウ段(u)音結尾。然後在不考慮其他特例的情況下，日文動詞大概可以分成以下5類，以辭書型表示：

- 五段動詞：以る(ru)以外的音結尾，或是以る結尾但是る前面是ア段(a)、ウ段(u)、オ段(o)音。例如：行く(iku)、取る(toru)
- 上一段動詞：以る(ru)結尾而且る(ru)前面是イ段(i)。例如：信じる(sinjiru)
- 下一段動詞：以る(ru)結尾而且る(ru)前面是エ段(e)。例如：食べる(taberu)
- サ行變格動詞：以する(suru)結尾的動詞，不屬於五段動詞。例如：勉強する(benkyousuru)
- カ行變格動詞：以来る(kuru)結尾的動詞，不屬於五段動詞。例如：来る(kuru)

注意到，以する(suru)結尾的動詞不屬於五段動詞，屬於サ行變格動詞、以来る(kuru)結尾的動詞不屬於五段動詞，屬於カ行變格動詞。

雞塊接著解釋，這些動詞變化為ます型的方式如下

- 五段動詞：將結尾的ウ段(u)改為イ段(i)，並加上ます(masu)。例如：行く(iku)->行きます(ikimasu)、取る(toru)->取ります(torimasu)
- 上一段動詞：將結尾的る(ru)移除並加上ます(masu)。例如：信じる(sinjiru)->信じます(sinjimasu)
- 下一段動詞：將結尾的る(ru)移除並加上ます(masu)。例如：食べる(taberu)->食べます(tabemasu)
- サ行變格動詞：將結尾的する(suru)改成します(simasu)。例如：勉強する(benkyousuru)->勉強します(benkyousimasu)
- カ行變格動詞：將結尾的来る(kuru)改成来ます(kimasu)。例如：来る(kuru)->来ます(kimasu)

以下為本題的日文平假名對照羅馬拼音表：

ア段(a)	イ段(i)	ウ段(u)	エ段(e)	オ段(o)	
あ(a)	い(i)	う(u)	え(e)	お(o)	
か(ka)	き(ki)	く(ku)	け(ke)	こ(ko)	
さ(sa)	し(si)	す(su)	せ(se)	そ(so)	
た(ta)	ち(ti)	つ(tu)	て(te)	と(to)	
な(na)	に(ni)	ぬ(nu)	ね(ne)	の(no)	
は(ha)	ひ(hi)	ふ(fu)	へ(he)	ほ(ho)	
ま(ma)	み(mi)	む(mu)	め(me)	も(mo)	
や(ya)		ゆ(yu)		よ(yo)	
ら(ra)	り(ri)	る(ru)	れ(re)	ろ(ro)	
わ(wa)				を(wo)	ん(n)

本題的輸入將會是由上表的羅馬拼音所組成的日文句子，並保證是以日文動詞的辭書型結尾。請根據以上資訊，將句子最後面的動詞改為ます型後輸出。保證所有測資的動詞變化都可以用以上資訊完成。

注：本題所述的日文動詞變化方法與實際日文動詞變化有所差異，請參賽者做題時以題目敘述為主。

輸入格式

第一行包含一個 T，代表接下來有 T 筆測資。

每筆測資有一行，包含一個字串 S ，即為題目所提到的輸入。

輸出格式

對於每筆測資，輸出符合題目要求的字串。

資料範圍

- $1 \leq T \leq 100$
- $1 \leq |S| \leq 100$，$|S|$ 代表字串 S 的長度

測試範例

輸入範例 1

```
3
taberu
okiru
sinjiru
```

輸出範例 1

```
tabemasu
okimasu
sinjimasu
```

輸入範例 2

```
5
aisiteiru
kyounaniwotaberu
yorunikakeru
bakuretusuru
okaesimousu
```

輸出範例 2

```
aisiteimasu
kyounaniwotabemasu
yorunikakemasu
bakuretusimasu
okaesimousimasu
```

範例說明

範例測資 1 中，`taberu`、`okiru`、`sinjiru` 皆為上一段或下一段動詞，ます型的變化方式為去掉 `ru` 並加上 `masu`。

範例測資 2 中：

- `aisiteiru` 是以 ru 結尾，且 ru 前面是 i，屬於上一段動詞，因此ます型應為 `aisiteimasu`
- `kyounaniwotaberu` 是以 ru 結尾，且 ru 前面為 e，屬於下一段動詞，因此ます型應為 `kyounaniwotabemasu`
- `yorunikakeru` 是以 ru 結尾，且 ru 前面為 e，屬於下一段動詞，因此ます型應為 `yorunikakemasu`
- `bakuretusuru` 是以 suru 結尾，屬於サ行變格動詞，因此ます型應為 `bakuretusimasu`
- `okaesimousu` 是以 ru 以外的音結尾，屬於五段動詞，因此ます型應為 `okaesimousimasu`

3_十一 (Eleven)

(15分)
時間限制: 1 second
記憶體限制: 256 MB

題目敘述

「在某些靈性和神秘主義的信仰體系中，11 被視為一個高度靈性的數字。它被認為是一個與靈性成長、直覺力和心靈覺醒相關的數字。」

「一些人相信 11 代表著一種共振或對宇宙的連接。它被視為一個提醒，使人們意識到他們與宇宙的聯繫，並且可以引導他們朝著更高的目標和目的前進。」

數字 11 在某些文化和信仰體系中具有特殊的含義。傳說中只要在今天晚上 11:11 分之前構造出一個滿足以下條件的數字，就能夠帶來好運：

- 這個數字沒有數碼 0。
- 這個數字的數碼和為 n。
- 這個數字可以被 11 整除。
- 這個數字是滿足以上三個條件中最大的數。

為了能夠得到好運，給定數字 n，請輸出滿足以上條件的數字，或回報不存在這個數字。

一個數字的數碼和為所有數碼的總和。舉例來說，123 的數碼和為 1 + 2 + 3 = 6。

輸入格式

輸入有一行，該行有一個正整數 n，意義與題目敘述相同。

輸出格式

請輸出一行，若沒有滿足題目所有條件的數字則輸出 `-1`，否則輸出該數字。

資料範圍

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

測試範例

輸入範例 1

```
2
```

輸出範例 1

```
11
```

輸入範例 2

```
3
```

輸出範例 2

```
-1
```

範例說明

可以證明當 $n = 3$ 時沒有滿足條件的數字。

4_路燈 (Street_Light)

(15分)
時間限制: 1 second
記憶體限制: 256 MB

問題敘述

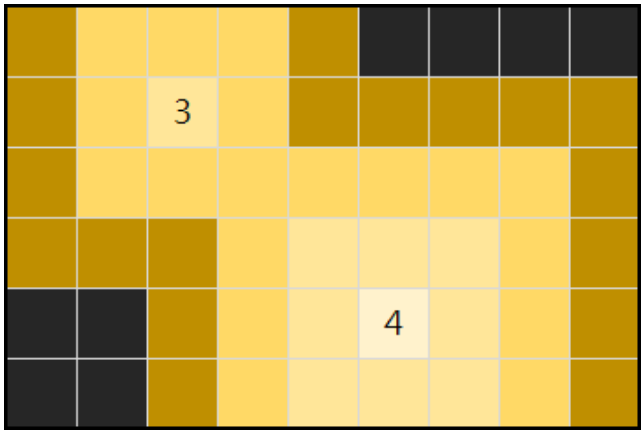
小庠是星星市的市長，今天他想在城市裡點亮一些路燈，照亮市民們的家門。

我們以一個 $N \times M$ 的表格表示星星市的地圖，小庠想在其中一些格子內設置路燈，每盞路燈可能有不同的強度。路燈所在的格子會有最高的亮度，並且隨著距離越遠，亮度會逐漸降低至 0。

更詳細的說，我們將第 r 行第 c 列的格子表示為 (r, c)。假設第 i 盞路燈設置於 (x_i, y_i) 且其強度為 p_i，則這盞路燈可以帶給格子 (r, c) 的亮度為 $\max(0, p_i - \max(|r - x_i|, |c - y_i|))$。

若有多盞路燈，則某個格子的亮度會是所有路燈給它的亮度的最大值。也就是說，假設總共設置了 K 盞路燈，格子 (r, c) 的亮度會是 $\max\limits_{1 \le i \le K}(\max(0, p_i - \max(|r - x_i|, |c - y_i|)))$。

以下圖為例，其城市大小為 6×9，左上角位於 $(2, 3)$ 的路燈強度為 3，右下角位於 $(5, 6)$ 的路燈強度為 4，該格顏色越淺表示其亮度越高，而深灰色的格子表示其亮度為 0。



小庠告訴你他希望星星市中每一個格子的亮度為何，你能寫一支程式幫小庠安排每一盞路燈設置的位置跟它們的強度嗎？

由於小庠很有錢，他不打算限制你使用的路燈的數量，但每個格子至多只能放一盞路燈。

輸入格式

輸入的第一行為兩個整數 N、M，表示星星市的大小。接下來 N 行，第 i 行包含 M 個整數 $b_{i,j}$，表示小庠希望星星市第 i 行第 j 列的格子的亮度為 $b_{i,j}$。

輸出格式

若不存在任何安排路燈的方式滿足小庠的要求，請輸出一行 -1。

否則，第一行請先輸出一個整數 K，表示欲設置的路燈數量。接下來 K 行，第 i 行包含三個整數 x_i、y_i、p_i，表示第 i 盞路燈的位於 (x_i, y_i) 且其強度為 p_i。

整數 K 必須滿足 $0 \leq K \leq N \times M$；路燈必須設置於合法的位置上，也就是說，$1 \leq x_i \leq N$ 且 $1 \leq y_i \leq M$；路燈的強度必須為不超過 1000 的正整數，也就是說，$1 \leq p_i \leq 1000$；任兩盞路燈的位置必須相異，也就是說，對於任意 $i \neq j$，應滿足 $(x_i, y_i) \neq (x_j, y_j)$。

你不需要最小化使用路燈的數量，且如果有多種可能的解，你可以輸出任意一種。

資料範圍

- $1 \leq N, M \leq 1000$。
- $0 \leq b_{i,j} \leq 1000$（$1 \leq i \leq N$、$1 \leq j \leq M$）。

範例

輸入範例 1

```
6 9
1 2 2 2 1 0 0 0 0
1 2 3 2 1 1 1 1 1
1 2 2 2 2 2 2 2 1
1 1 1 2 3 3 3 2 1
0 0 1 2 3 4 3 2 1
0 0 1 2 3 3 3 2 1
```

輸出範例 1

```
2
2 3 3
5 6 4
```

輸入範例 2

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

輸出範例 2

```
-1
```

範例說明

對於範例 1，請參考問題敘述中的圖片。

對於範例 2，不存在設置路燈的方式滿足小庠的要求。

5_遊戲體力值 (Game_Stamina)

(15分)
時間限制: 1 second
記憶體限制: 256 MB

問題敘述

在遊戲 YTP（Young Turing Protection）中，玩家可以透過一場又一場的防守戰從敵人手中保護少年圖靈，來享受刺激的遊戲體驗，但為了給予玩家一定程度的限制，玩家每次參與一場防守戰都必須消耗「體力值」，而這個體力值每過 k 分鐘才會增加一點，因此，玩家時常會需要等待時間過去才有辦法賺取到足夠的體力值繼續遊玩遊戲。

沈迷於 YTP 的小 Y 在今天已經玩了若干場防守戰，但接下來有一段時間他將無法長時間的遊玩一場防守戰，只好偶爾查看一下遊戲的當前體力值，好計算等他有空時他可以遊玩幾場防守戰。

不過，即使知道這遊戲會每 k 分鐘會增加一點體力值，小 Y 卻不知道具體來說是在哪幾分鐘時會增加一點。舉例來說，若 $k = 3$，體力值有可能會在第 $3, 6, 9, 12, \ldots$ 分鐘、或是 $1, 4, 7, 10, \ldots$ 分鐘、又或者是 $2, 5, 8, 11, \ldots$ 分鐘時增加一點體力值。也就是說，肯定有一個餘數 $0 \leq r < k$，滿足只要過了 t 分鐘，且 t 除以 k 的餘數是 r 時就會增加一點體力值。

已知小 Y 在接下來的時間內，做了 N 次體力值的檢查，其中第 i 次檢查在第 t_i 分鐘，且體力值為 s_i，而這之間小 Y 都沒有消耗過任何體力值，請你在每次他檢查後，根據到目前為止他見過的體力值，對於每個餘數，輸出他有沒有可能是 r。

輸入格式

第一行包含兩個整數 N、k，代表小 Y 的檢查次數和體力值增加的時間間隔。

接下來有 N 行，其中第 i 行包含兩個整數 t_i、s_i，表示小 Y 第 i 次檢查時是在第 t_i 分鐘，且當時的體力值為 s_i。

輸出格式

對於第 i 次檢查，輸出一行一個 01 字串 $p_0p_1p_2\ldots p_{k-1}$，p_j 是 1 代表餘數 j 是一個可能的 r，0 則代表不可能。該字串表示到第 i 次檢查為止，可能的餘數 r 是哪些。

資料範圍

- $1 \leq N, k \leq 1000$。
- $1 \leq t_i, s_i \leq 10^9$ $(1 \leq i \leq N)$ 。
- $t_1 < t_2 < \cdots < t_N$。
- 保證存在至少一個餘數 r 滿足所有的輸入，意即不會有矛盾的情況發生。

範例

輸入範例 1

```
5 3
1 1
3 2
6 3
8 3
11 4
```

輸出範例 1

```
111
101
101
100
100
```

輸入範例 2

```
10 10
3971 82568
35370 85708
39553 86126
43370 86508
60282 88199
63915 88562
65994 88770
77566 89927
81695 90340
96803 91851
```

輸出範例 2

```
1111111111
1011111111
1000111111
1000111111
1000111111
1000001111
1000001111
1000000111
1000000111
1000000111
```

範例說明

輸入範例 1 的五次檢查分別對應如下：

- 小 Y 在第 1 分鐘進行了他的第一次檢查，其體力值為 1，由於資訊尚未足夠，因此所有的餘數都是可能的 r。
- 小 Y 在第 3 分鐘進行了他的第二次檢查，其體力值為 2，若 $r = 1$，那麼時間 $1, 4, 7, \ldots$ 才會增加體力值，因此第 3 分鐘的體力值不該增加，1 不是一種合理的餘數 r。
- 小 Y 在第 6 分鐘進行了他的第二次檢查，其體力值為 3，並無任何新發現。
- 小 Y 在第 8 分鐘進行了他的第二次檢查，其體力值為 3，若 $r = 2$，那麼時間 $2, 5, 8, \ldots$ 肯定會增加體力值，因此第 8 分鐘的體力值必須增加，2 不是一種合理的餘數 r。
- 小 Y 在第 11 分鐘進行了他的第二次檢查，其體力值為 4，並無任何新發現。

輸入範例 2 僅用作一個輸入大小稍大的測試資料供選手自我測試用。

6_漆彈遊戲 (Paintball_Game)

(15分)
時間限制: 1 second
記憶體限制: 256 MB

問題敘述

最近竹子非常熱衷一款漆彈遊戲，這個遊戲在一個 $N \times N$ 的棋盤上進行，而且一共有 M 名玩家。遊戲中，玩家會被編號為 1 到 M、玩家們會按照編號從 1 到 M 依序進行操作，其中第 i 名玩家操作時，會在棋盤上 (x_i, y_i) 的位置引爆一顆爆炸強度為 l_i 的漆彈炸彈，引爆後會將引爆位置的十字方向上、寬度為 l_i 的方格都染上顏色 i，會覆蓋方格上原有的顏色。也就是說，滿足 $\min(|x' - x_i|, |y' - y_i|) \leq \frac{l_i - 1}{2}$ 的所有格子 (x', y') 的顏色都會變成 i，而這些被染上顏色 i 的方格被稱為玩家 i 的「領地」。

當所有玩家都做完操作後遊戲便會結束，此時擁有最多領地的玩家會獲得勝利。為了時時刻刻掌握戰局，在第 i 回合結束時，竹子會想要在 (a_i, b_i)、(c_i, d_i) 這兩個方格所形成的矩形區域中計算玩家 p_i 的領地數量。由於遊戲每次的決策都有時間限制，因此他希望可以用一支程式來快速的計算他想知道的資訊，聰明的你願意幫助竹子完成他的夢想嗎？

輸入格式

輸入的第一行包含兩個正整數 N、M，代表棋盤的大小、遊戲人數。

接下來有 M 組輸入，每組包含兩行，代表一個回合內的操作。

其中第 i 組的第一行包含三個整數 l_i、x_i、y_i 分別代表第 i 位玩家引爆的漆彈炸彈強度以及位置。而第 i 組的第二行包含五個整數 p_i、a_i、b_i、c_i、d_i 代表回合結束時，竹子想調查的玩家代號、以及調查範圍的兩個端點。

輸出格式

輸出 M 行，每行輸出一個整數，其中第 i 行的輸出代表第 i 回合結束後，竹子想調查的玩家 p_i 在指定範圍內的領地總數。

資料範圍

- $1 \leq N, M \leq 5000$。
- $1 \leq x_i, y_i \leq N$（$1 \leq i \leq M$）。
- $1 \leq l_i \leq N$（$1 \leq i \leq M$）。
- l_i 是奇數（$1 \leq i \leq M$）。
- $1 \leq a_i \leq c_i \leq N$（$1 \leq i \leq M$）。
- $1 \leq b_i \leq d_i \leq N$（$1 \leq i \leq M$）。
- $1 \leq p_i \leq M$（$1 \leq i \leq M$）。

範例

輸入範例 1

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

輸出範例 1

```
5
3
7
```

輸入範例 2

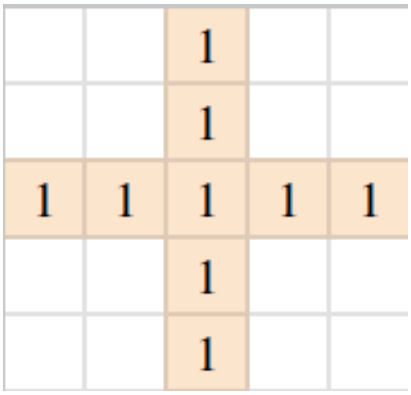
```
6 4
5 2 1
3 4 5 5 6
1 3 2
1 3 1 5 4
3 3 2
2 1 1 6 6
3 6 6
1 1 2 3 6
```

輸出範例 2

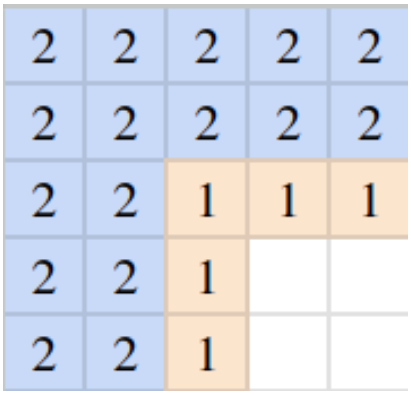
```
0
5
0
1
```

範例說明

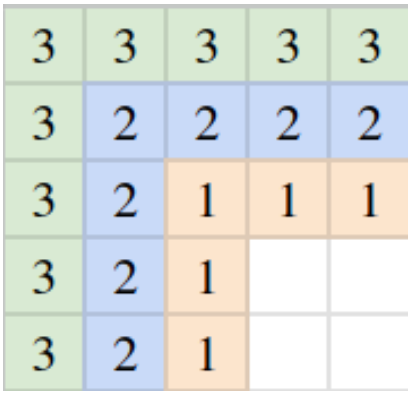
在範例 1 中，三名玩家操作後的版面分別如下圖所示：



第一名玩家操作後

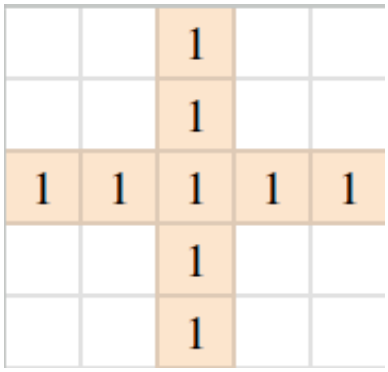


第二名玩家操作後

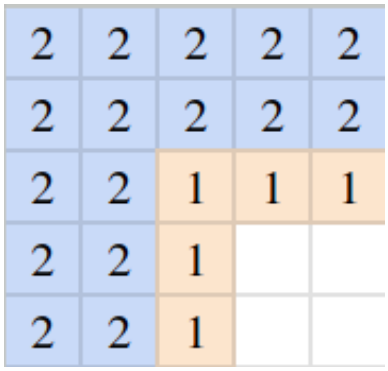


第三名玩家操作後

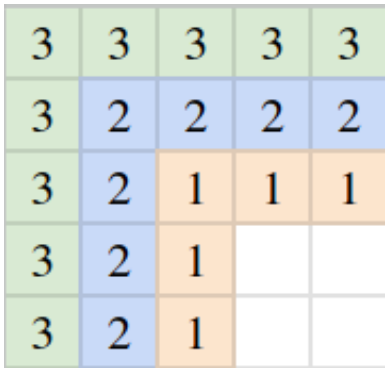
第一名玩家操作後：



第二名玩家操作後：



第三名玩家操作後：



7_收費站模擬器 (Toll_Plaza_Simulator)

(15分)
時間限制: 3 seconds
記憶體限制: 512 MB

問題敘述

VVivvi 最近沉迷於卡車模擬器，然而玩了沒多久，他便發現他不太擅長開卡車。「不如我來玩收費站模擬器好了。」因為開錯車道而卡在路中間的他這麼想著。

這個遊戲非常簡單：在一條單向公路上的某處有一個收費站，這個收費站有 N 條車道，每條車道都有一個收費亭，負責向這個車道上經過收費站的車收費。這些收費亭和車道由左至右編號為 $1, 2, \ldots, N$。這個收費站在晚上總是沒有任何車經過，但是一天亮就會忽然有許多車出現。VVivvi 已經預料到明天早上日出時，每一條車道都會有恰好一輛車出現。

本來這是個很簡單的遊戲，但是有時會發生突發狀況：有些收費亭會因為設備故障無法運作。如果有車輛的駕駛發現，他所在車道的收費亭是關閉的，那麼他就會把車開到距離最近且收費亭有運作的車道，如果有兩個收費亭有運作的車道距離相同，他會選擇編號較小的那個。兩條車道的距離為它們編號差的絕對值。這個遊戲的目標就是要為每個收費亭分配人力，避免有員工過勞或特別閒的狀況出現。

現在是夜晚，一開始所有收費亭都是正常運作的，接下來在天亮之前會有 Q 個事件，每個事件是以下其中一種：

- $1\ \ \ell\ \ r$：對於所有滿足 $\ell \leq i \leq r$ 的 i，第 i 座收費亭的設備故障了。注意第 i 座收費亭有可能本來就是故障的。
- $2\ \ \ell\ \ r$：對於所有滿足 $\ell \leq i \leq r$ 的 i，第 i 座收費亭可以正常運作。注意第 i 座收費亭有可能本來就可以正常運作。

雖然一切都發生在日出之前，不會有任何車經過的時候，但 VVivvi 希望可以在每次事件發生之後，預測如果目前所有收費站的狀態都維持不變到天亮，日出那一刻的混亂程度。日出時的混亂程度定義為那時出現的所有車輛的總移動距離，假設一輛車從第 i 條車道來、在第 j 個收費亭繳費，那麼它的移動距離是 $|i - j|$。

第 1 和第 N 個收費亭因為最靠近邊邊，VVivvi 可以輕易管理，所以不會發生任何事件。也就是說，這兩個收費亭永遠都是開啟的。

輸入格式

第一行包含兩個整數 N、Q，表示車道和事件的數量。

接下來有 Q 行，其中第 i 行包含三個整數 t、ℓ、r，表示第 i 個事件的類型和發生的範圍。

輸出格式

輸出 Q 行，其中第 i 個包含一個整數，表示第 i 個事件過後，預測的日出時的混亂程度。

資料範圍

- $3 \leq N \leq 10^6$。
- $1 \leq Q \leq 10^6$。
- $t \in \{1, 2\}$。

- $1 < \ell \leq r < N$。

範例

輸入範例 1

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

輸出範例 1

```
2
6
4
10
20
```

輸入範例 2

```
1000000 10
1 21586 928842
1 619334 853861
2 274417 572426
1 239830 413748
2 79123 839131
1 2410 978513
2 350282 605034
1 22256 122739
1 463714 491706
1 588774 892375
```

輸出範例 2

```
205779269641
205779269641
47739274528
70206420196
2839716097
238195242756
65125733632
65125733632
65321649641
68424333811
```

範例說明

範例 1 中，每個事件的說明如下：

1. 第 2 和 3 個收費亭關閉了，因此第 2 個車道的車會改到第 1 個收費亭、第 3 個車道的車會改到第 4 個收費亭，其餘的車會到本來的車道的收費亭。
2. 第 4 和 5 個收費亭關閉了，現在第 2 到 5 個收費亭是關閉的，這些車道的車分別會到第 $1, 1, 6, 6$ 個收費亭。
3. 第 $5, 6, 7$ 個收費亭開啟了，現在第 2 到 4 個收費亭是關閉的，這些車道的車分別會到第 $1, 1, 5$ 個收費亭。
4. 第 6 到 9 個收費亭關閉了，現在只有第 $1, 5, 10$ 個收費亭是開啟的，每個車道的車分別會到第 $1, 1, 5, 5, 5, 5, 5, 10, 10, 10$ 個收費亭。
5. 第 4 到 6 個收費亭關閉了，現在只有第 1 和 10 個收費亭是開啟的，第 1 到 5 個車道的車會到第 1 個收費亭，其餘會到第 10 個收費亭。

8_Ice_Skating

(4分 /16分)
時間限制: 3 seconds
記憶體限制: 512 MB

問題敘述

身為一個寶可夢愛好者，你喜歡到處去道館挑戰，有一天你得知了寶可夢大師小智開了一家全新的道館，你迫不及待要去挑戰看看。

你到達了這座道場，卻發現了一個問題：你找不到挑戰入口在哪裡。這個道場其實是一個 $N \times M$ 的溜冰場，挑戰入口在其中的一個格子內。在溜冰場內，你只能用以下方式來移動：

1. 選擇一個方向（上、下、左、右）。
2. 向所選方向前進，直到撞到障礙物才能停止。

一開始你會在溜冰場上標記為 `S` 的方格上，請你找到到達挑戰入口 `E` 的最少移動次數。如果無法到達挑戰入口則輸出 -1。

請注意：溜冰場外視為障礙物，且只要碰到挑戰入口（不用停在入口）即可算到達。

輸入格式

第一行輸入兩個正整數 N、M，代表溜冰場的長、寬。

接下來 N 行，每行有 M 個字元 $s_{i,1}, s_{i,2}, \ldots, s_{i,M}$，每個字元為 `#.SE` 其中之一。

- `#` 代表障礙物。
- `.` 代表空白格子。
- `S` 代表你所在的起始點。
- `E` 代表道館的挑戰入口。

輸出格式

輸出一個整數，代表到達挑戰入口的最少移動次數。如果無法到達挑戰入口則輸出 -1。

資料範圍

- $2 \leq N, M \leq 100\,000$。
- $N \times M \leq 1\,000\,000$。
- $s_{i,j} \in \{\texttt{\#}, \texttt{.}, \texttt{S}, \texttt{E}\}$（$1 \leq i \leq N$、$1 \leq j \leq M$）。
- 保證溜冰場上的 $\texttt{S}$ 跟 $\texttt{E}$ 都恰有一個。

子任務

- 子任務 1 (4 分) $N \times M \leq 10\,000$。

- 子任務 2 (16 分) 無額外限制。

範例

輸入範例 1

```
5 5
S....
.....
.....
.....
....E
```

輸出範例 1

```
2
```

輸入範例 2

```
5 5
S.#..
.....
#.###
.....
..E..
```

輸出範例 2

```
3
```

輸入範例 3

```
2 2
S#
#E
```

輸出範例 3

```
-1
```

輸入範例 4

```
4 4
....
.S..
..E.
....
```

輸出範例 4

```
-1
```

範例說明

在範例 1 中，其中一種移動方法是先往右再往下。顯然不存在一步就能移動到挑戰入口的方法。

在範例 2 中，唯一可以在三步以內移動到終點的方法是右、下、右。雖然最後一步不會讓你停在挑戰入口處，但只是經過挑戰入口也會算是到達。

在範例 3 中，你無法進行任何移動。

在範例 4 中，請注意你只能在撞到障礙物後停下來。

9_Rectangles

(2分 /3分 /15分)
時間限制: 3 seconds
記憶體限制: 512 MB

問題敘述

在一個 $n \times n$ 的格子圖 G 上，有些格子中有障礙物，有些沒有。

請問有幾個不包含任何障礙物的矩型。

意即，在這個 $n \times n$ 的二維矩陣裡，有多少個二維子陣列使得其中沒有任何障礙物。

輸入格式

輸入的第一行會包含一個正整數 n。

接下來的 n 行，每行有 n 個字元 $G_{i,0}, G_{i,1}, \ldots, G_{i,n-1}$，分別代表該位置有沒有障礙物。

如果有，該字元會是 '#'，如果沒有障礙物則會是 '.'。

輸出格式

請輸出一行包含一個非負整數，代表有多少個矩型，使得這個矩型不包含任何障礙物。

資料範圍

- $1 \leq n \leq 5000$。
- $G_{i,j} \in \{\#, .\}$ $(0 \leq i, j \leq n-1)$ 。

子任務

- 子任務 1 (2 分) $1 \leq n \leq 50$。
- 子任務 2 (3 分) $1 \leq n \leq 200$。
- 子任務 3 (15 分) 無額外限制。

範例

輸入範例 1

```
4
...#
#..#
..#.
.#.#
```

輸出範例 1

```
21
```

輸入範例 2

```
5
.#.#.
#...#
.##.#
###..
..#.#
```

輸出範例 2

```
25
```

範例說明

範例1中：

1x1的矩型有10個。

1x2的矩型有4個。

1x3的矩型有1個。

2x1的矩型有4個。

2x2的矩型有1個。

共有21個矩型。

10_吃蛋糕遊戲 (Eating_Cake)

(5分 /20分)
時間限制: 3 seconds
記憶體限制: 512 MB

問題敘述

俄羅斯輪盤章魚燒是一個知名的遊戲，具體玩法是在一些章魚燒中，摻雜一個放入很辣的配料，並讓參與者隨機挑選章魚燒並吃下。

另外，蛋糕作為著名的甜點受到大眾的喜愛，然而，太甜有時後也會讓人受不了。

因此，結合俄羅斯輪盤章魚燒與蛋糕，就誕生出了一款新的遊戲 --- 吃蛋糕遊戲。

具體如下：

遊戲中有兩個玩家 A、B，並各自有著甜度生命值 LA、LB。

一開始會有一個 $N \times M$ 棋盤狀的蛋糕，對於蛋糕上的每一格，其甜度可能會太高、或者是適中。

玩家每吃到一塊甜度太高的格子，甜度生命值就會扣一。

在遊戲中，兩名玩家會輪流行動，由 A 先開始。每次會挑選一個尚未被吃掉的格子 (i, j)，並以其作為左上角，將其右下角的所有蛋糕切下來吃掉，也就是將所有滿足 $i \leq x \leq N$ 且 $j \leq y \leq M$ 並且尚未被吃掉的 (x, y) 格子吃掉。

如下圖所示，若選擇了 $(i, j) = (2, 2)$ 這一格當作左上角，那麼圖中標為灰色的那些格子也必須一併被吃掉。

(1, 1)	(1, 2)	(1, 3)	(1, 4)
(2, 1)	(2, 2)	(2, 3)	(2, 4)
(3, 1)	(3, 2)		

當一位玩家的甜度生命值歸零、或降為負數時，這位玩家就會輸掉遊戲。

給定 $N \times M$ 的蛋糕，與其上的每一格的甜度是否太甜，並給定 A、B 的甜度生命值，請判斷哪位玩家有必勝策略。

輸入格式

輸入的第一行包含一個正整數 Q，代表總共有幾筆測資。

每筆測資的第一行包含兩個正整數 N、M，代表蛋糕的大小。

對於接下來的 N 行中，每行會包含 M 個整數 $x_{i,1}, x_{i,2}, \ldots, x_{i,M}$。

$x_{i,j} = 1$ 代表蛋糕上 (i, j) 格的甜度太高；$x_{i,j} = 0$ 則代表蛋糕上 (i, j) 格的甜度沒有太高（適中）。

每筆測資的最後一行包含兩個正整數 LA、LB，代表玩家 A 和玩家 B 的甜度生命值。

輸出格式

對於每筆測資，若玩家 A 有必勝策略，請輸出 `A`；若玩家 B 有必勝策略，請輸出 `B`；否則請輸出 `Tie`。

資料範圍

- $1 \leq Q \leq 300$。
- $1 \leq N, M$。
- $1 \leq N \times M \leq 400$。
- $x_{i,j} \in \{0, 1\}$ $(1 \leq i \leq N、1 \leq j \leq M)$ 。
- $1 \leq LA, LB \leq N \times M$。

子任務

- 子任務 1 (5 分) $N = 1$。
- 子任務 2 (20 分) 無額外限制。

範例

輸入範例 1

```
3
3 3
0 1 0
0 0 0
1 0 1
3 1
2 3
1 1 1
0 1 1
1 2
1 1
0
1 1
```

輸出範例 1

```
A
B
Tie
```

範例說明

第一筆測資中，A 只要在第一回合將右邊兩排全部吃下，就能夠強迫 B 一定要吃掉 $(3,1)$ 的太甜格子，從而導致 B 的甜度生命值歸零而落敗。

在第二筆測資中，A 在第一回合時一定得吃掉 $(2,3)$ 的太甜格子，從而導致甜度生命值歸零而落敗。

在第三筆測資中，A 在第一回合吃掉 $(1,1)$ 的格子後，蛋糕就全被吃光了，因此沒有人的甜度生命值會歸零、或降為負數，所以結果是平手。