2011 網際網路程式設計全國大賽
高中組決賽

- 題目：本次比賽共八題（含本封面共 28 頁）。
- 題目輸入：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，依題目敍述分隔。
- 題目輸出：全部的輸出皆輸出到螢幕(**標準輸出**)。
- 時間限制：每一題的執行時間限制如下表所示。其間執行的電腦上不會有別的動作、也不會使用鍵盤或滑鼠。
- 比賽中上傳之程式碼請依照以下規則命名:
  1. 若使用 C 做爲比賽語言則命名爲 `pa.c`, `pb.c`, 以此類推
  2. 若使用 C++ 做爲比賽語言則命名爲 `pa.cpp`, `pb.cpp`, 以此類推

  未按照此規則命名之程式碼將可能因此得到 Compilation Error。
- `cin` 輸入經測試發現速度遠慢於 `scanf` 輸入，答題者若使用需自行承擔因輸入速度過慢導致 Time Limit Exceeded 的風險。

表 1: 題目資訊

	題目名稱	執行時間限制
題目**A**	三角形金磚	5 秒
題目**B**	玄鐵X金輪	15 秒
題目**C**	破解密碼	10 秒
題目**D**	河川改道之術	5 秒
題目**E**	整修中的道路	10 秒
題目**F**	田忌賽馬	5 秒
題目**G**	上帝 GOD	10 秒
題目**H**	圖靈機	10 秒

2011 網際網路程式設計全國大賽
解題程式輸入輸出範例

C 程式範例：

```
#include <stdio.h>
int main(void){
    int cases, i;
    double a, b;
    scanf("%d", &cases);
    for(i = 0;i < cases;i++){
        scanf("%lf %lf", &a, &b);
        printf("%.2f\n", a+b);
    }
    return 0;
}
```

C++ 程式範例：

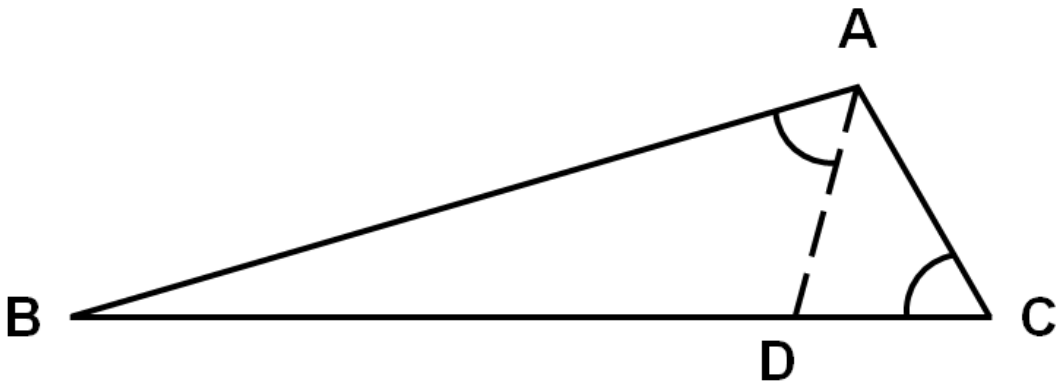
```
#include <iomanip>
#include <iostream>
using namespace std;
int main(void){
    int cases, i;
    double a, b;
    cin >> cases;
    cout << fixed << setprecision(2);
    for(i = 0;i < cases;i++){
        cin >> a >> b;
        cout << a+b << endl;
    }
    return 0;
}
```

題目 A
三角形金磚

執行時間限制: 5 秒

塔克拉瑪干沙漠的別名是「死亡之海」，身爲考古探險隊隊長的亞特想要聘請一位嚮導，幫助考古探險隊展開在「死亡之海」的旅程。他上次在塔克拉瑪干沙漠考察的時候認識了一個嚮導，這個嚮導自稱能夠帶領亞特前往傳說中的沙漠古城，條件是從沙漠古城帶出來的部份寶藏要分給嚮導。

根據嚮導所言，這個沙漠古城中的寶藏都是厚度一致的鈍角三角形金磚，而且兩個銳角必定不相等。爲了方便起見，我們將金磚的鈍角稱爲 A 、比較小的銳角稱爲 B 、比較大的銳角稱爲 C ，再把金磚從鈍角 A 分爲兩個三角形，並且將新的頂點稱爲 D ，分割金磚的條件是讓 $\angle BAD = \angle ACD$。嚮導要求的報酬就是和 $\triangle ACD$ 金磚等價的金錢或寶物。



雖然亞特並不是那麼相信嚮導口中的沙漠古城，但是亞特想要探險的地方風險實在太大，沒有其他嚮導願意陪同，嚮導人選因此決定。因爲亞特不太相信會從這趟旅程帶回傳說中的寶物，所以他不想花時間計算嚮導要求的報酬佔了所有寶物的多少比例，你能代替他算出來嗎？

■ 輸入檔說明

第一行有一個整數 T $(T \leq 100)$，表示接下來有幾組測試資料。

每組測試資料包含兩個整數，分別代表 $\overline{AB}$ 的長度和 $\overline{BC}$ 的長度。$(1 < \overline{AB} < \overline{BC} < 100, \overline{BC} < 2\overline{AB})$

■ 輸出檔說明

對於每組測試資料輸出一個浮點數，表示 $\triangle ACD$ 佔 $\triangle ABC$ 的比例，四捨五入至小數第三位。

■ 範例輸入

```
3
4 6
12 14
24 26
```

■ 範例輸出

```
0.556
0.265
0.148
```

題目 B
玄鐵X金輪

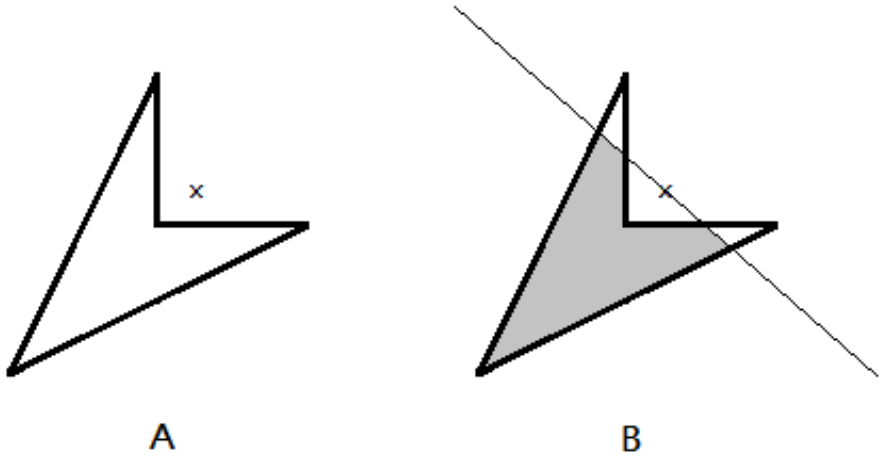
執行時間限制: 15 秒

在與蒙古軍對戰一役中，神雕大俠楊過與金輪國師激鬥甚久，雙方拆至三百多招在江湖中實屬鮮見，而這其中的原因更是個令人津津樂道的故事。

其實楊過經過一十八年的修練，功力早就無人能敵，甫拆至三十招就已經胸有成竹，立於不敗之地。但此時他心中甚是掛念敵人手中那金輪，那純金打造的精細雕工，即使直接拿去當鋪典當也有好幾百兩。再看看手上的玄鐵重劍，有如路邊撿來的破銅爛鐵，心中不免感到不快。

在先前的酣鬥中，金輪國師的要害早已暴露無遺，但都被他用金輪護住了，即使玄鐵重劍威力巨大能夠在一招之內擊碎金輪並打中要害，楊過也不敢下手。猶如早已知道楊過心中所想，故反其道而行。苦戰良久楊過略顯力竭，情急之下楊過決定下一招就要打倒金輪國師並保留最多殘餘的金輪。

楊過使用玄鐵重劍發出的招式是一條直線，並且會摧毀直線某一邊的物體。金輪是由一個簡單多邊形表示，而金輪國師的要害則以 X 表示，如圖 A。如果楊過想要擊中金輪國師的要害並摧毀最少面積的金輪，可以用如圖 B 的方式發招。如此圖 B 中灰色的部分即可完全保留下來，而白色的部分則被玄鐵重劍摧毀。



你的任務是幫助楊過計算最多能保留下多大面積的金輪。不管楊過心中如何打算，起碼在當鋪典當時金輪的價錢是以面積計算的。

■ 輸入檔說明

第一行有一個整數 T $(T \leq 20)$，代表接下來有幾組測試資料。

每組測試資料以一個整數 N 和兩個浮點數 X, Y 開頭 $(1 \leq N \leq 1000, -1000 \leq X, Y \leq 1000)$，$N$ 代表金輪可以用幾個點表示，X 和 Y 則表示金輪國師要害的位置。接下來 N 行以順時針順序給定描述金輪形狀的點，每個點都以兩個整數 x_i 和 y_i 表示。$(-1000 \leq x_i, y_i \leq 1000)$

請注意，金輪國師的要害可能在金輪上或是在金輪外，或是與金輪的某條邊或某個點重合。

■ 輸出檔說明

對於每一筆測試資料請輸出一個浮點數，表示最多有多大面積的金輪可以被保留下來，請四捨五入至小數第六位。

■ 範例輸入

```
2
3 0 0
0 1
1 1
1 0
4 0.2 0.2
0 0
1 0
-1 -1
0 1
```

■ 範例輸出

```
0.500000
0.882154
```

題目 C
破解密碼

執行時間限制: 10 **秒**

密碼學是研究如何隱密地傳遞訊息的科學，從古代的戰爭到現代電腦和網路通訊上有著非常重要的應用。相信你聽過「凱薩密碼」，它可以說是最經典的加密技術。凱薩密碼屬於替換式密碼，也就是把一個字母替換成另一個字母。我們稱加密前的文字爲「明文」，加密後的爲「密文」。一種凱薩密碼的替換方式是 A→D, B→E, ···, X→A, Y→B, Z→C。以數學的方式來說，我們讓 A=0, B=1, ···, Z=25，使用密碼 x 加密字母 a 得到的密文是

$$E_x(a) = (a + x) \mod 26$$

以現代的觀點，凱薩密碼是很容易破解的。即使我們不知道其中的密碼 x，我們也可以透過頻率分析的技術來反推密碼。以一般的英文來說，E 和 T 出現的機率最高，Q 和 Z 出現的機率很低。利用這個特點，我們只要統計密文中各個字母出現的頻率，就可以猜出密碼 x。更進一步，我們可以分析兩個字母連續出現的頻率，像是 TH 和 ER 這種的出現頻率就應該比較高。

這裡我們介紹一個新的加密方法，是凱薩密碼的改良版。你有一組密碼 $x_1x_2\cdots x_l$，要加密一組明文 $a_1a_2\cdots a_n$ 得到密文 $c_1c_2\cdots c_n$。我們用第一個密碼字母 x_1 加密 a_1，用 x_2 加密 a_2，···。如果密碼用完了，就拿前面得到的密文來用，也就是用 c_1 加密 a_{l+1}，···。舉例來說，用密碼 BEE 加密明文 CAKES 變成密文 DEOHW。

$$c_i = \begin{cases} a_i + x_i \mod 26 & \text{若 } i \le l \\ a_i + c_{i-l} \mod 26 & \text{若 } i > l \end{cases}$$

	C	A	K	E	S
+	B	E	E	D	E
	D	E	O	H	W

現在，給你幾組用同樣的密碼加密的明文密文配對，請你寫程式猜出密碼。

■ 輸入檔說明

第一行有一個整數 T ($T \leq 200$)，代表接下來有幾組測試資料。

每一組測試資料的第一行有一個數字 N ($N \leq 8$)，代表有幾組明文密文配對。

接下來 N 行每行有兩個字串，前面的代表明文，後面的是對應的密文。

每一行裡的明文和密文一定一樣長，長度介於 1 和 30000 中間。明文和密文都只有大寫英文字母，中間以一個空白隔開。

■ 輸出檔說明

對每一組測試資料輸出一個字串，代表能夠把 N 組明文分別加密成對應的密文的密碼。

如果有多組可能的密碼，請輸出最短的一個。如果沒有任何符合的密碼，則輸出 "-"。（密碼至少有一個字）

■ 範例輸入

```
3
2
CAKES DEOHW
CAKES DEOHW
2
CAKES DEOHW
CAKES CAKES
2
ABCD NQUF
ABCDE NQUFR
```

■ 範例輸出

```
BEE
-
NPSC
```

題目 D
河川改道之術

執行時間限制: 5 **秒**

歪歪國的河川都是歪歪的。歪歪國的人們每次出外旅行的時候總是會讓其他人留下深刻印象，其他人會說：啊哈，我會記得你們這些歪國人。

不過，歪歪國的人民很自豪地說，我們是歪中有序。儘管所有的河川都是歪歪扭扭的，但國土可是方方正正，而且可以分成 $M \times N$ 個小正方形區域，每一個小正方形區域都恰有兩條河道經過。更有趣的是，如果只看這兩條河道經過的方式，會發現其實只有兩種形態：

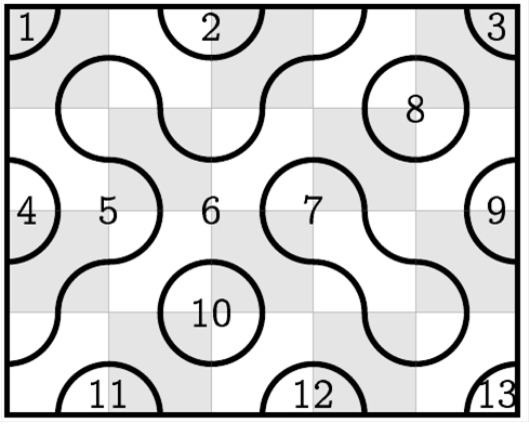
(a) A形態

(b) B形態

而實際的國土例子可能是：

```
ABAAB
BBABA
AABBB
ABABA
```

(a) 國土的顯示範例



(b) 範例對應的實際樣貌

歪歪國由於河道衆多，所有土地可能會被河道切成了很多塊。自古以來，歪歪國便依照國土分塊的狀況制定行政區域，如上圖右邊的數字編號。換句話說，只要國土上任兩點之間存在一條路線連結完全落在國土內，而且該路線不跨越任何河道，它們就會屬於同一個行政區域。

不過，由於當前的行政區域過於衆多，歪歪國想要改變河道行進的方式，使得以相同定義新劃分的行政區域總數降至最低。

史歪哩(Swyly)和科西(Kshi)是歪歪國裡面著名的魔法師，他們被歪歪國的水利大臣球球(Kel-kel)和皮皮(Pi-pi)聘請來整治歪歪國的國土。經過了許久的研究，爲了不打破歪歪國的國土風格，他們決定分次施以控制力(Ctrl-Z)，每次可將一塊 A 形態的小方格上面的水道轉換成 B 形態的小方格，或將 B 形態的小方格轉換成 A 形態的小方格。

「這，究竟給不給力啊？」歪歪國的審查大臣文文(Vim-vim)歪著頭，看著魔法師提出的國土修正方案。

爲了省力，史歪哩和科西打算用最少的施力次數，使得新劃分出來的行政區域數量降至最低。不僅如此，他們還希望找出盡可能美觀的方式，所謂盡可能地美觀，就是將修正後的國土每個小方格的形態寫出來以後一橫列一橫列接成一個字串時，是所有可能施以最少次控制力的做法當中，字典順序最小的一個方案。

時候不多了，趕快幫兩位魔法師找出整治歪歪國的最佳方案吧！

■ 輸入檔說明

輸入的第一行有一個正整數 T ($T \leq 100$) 表示測試資料的數量。每一筆測試資料的第一行有兩個正整數 M, N ($1 \leq M, N \leq 500$) 表示歪歪國的國土長寬，接下來的 M 行每一行有一個長度爲 N 的字串，第 i 行的第 j 個字元代表的就是 (i, j) 這一格的河道分布形態。($1 \leq i \leq M; 1 \leq j \leq N$)

■ 輸出檔說明

對於每一筆測試資料，第一行請輸出兩個以一個空白隔開的非負整數 R, K，其中 R 表示所能夠達到的最低行政區域數量，而 K 表示至少要進行幾次名爲控制力的河川改道之術。由於直接輸出最佳方案資料量太過巨大，我們採用下面的方式將答案雜湊後輸出，此函式與解題無關，純粹是方便輸出使用：

我們首先定義答案矩陣 A 爲：

以 M 列每一列有 N 個字元用以表示盡可能美觀的最佳方案。若第 (i, j) 格不需要施力，以點 '.' 表示；若需要施力，則以星號 '*' 表示。

現在我們把答案矩陣 A 的每一列依序接成一個長度恰好是 $M \times N$ 的字串 S，然後利用以下的函式跑出我們要的答案：

```
unsigned int get_hash(const char *S) {
    unsigned int i, h = 1315423911;
    for (i = 0; *S; S++, i++)
        h ^= ((h << 5) + (*S) + (h >> 2));
    h &= 0x7FFFFFFF;
    return h;
}
```

請你將這個函式回傳的值在每筆測資的第二列輸出。

■ 範例輸入

```
3
3 5
AAAAA
AAAAA
AAAAA
2 2
AB
BA
4 5
ABAAB
BBABA
AABBB
ABABA
```

■ 範例輸出

```
9 0
1804696164
5 1
1131664733
10 3
91716038
```

■ 範例輸出說明

第一筆範例測試資料的答案矩陣長得像這樣：

```
.....
.....
.....
```

第二筆範例測試資料的答案矩陣長得像這樣：

```
.*
..
```

第三筆範例測試資料的答案矩陣長得像這樣：

```
....*
...*.
..*..
.....
```

題目 E
整修中的道路

執行時間限制: 10 **秒**

卡恩國是一個非常現代化的國家，其中不乏許多國際知名的大都市，盛名享譽全球。而且，卡恩國不僅僅是先進的科技大國，由於其環境綠化執行得徹底以及國民們都具備良好的環保意識的緣故，卡恩國境內也充滿著許許多多著名的風景區，保有著許多工業大國所沒有的青山綠水。此外，卡恩國也引領著世界時尚和娛樂的潮流，「無煙囪工業」亦十分發達。綜合以上種種的因素，使得卡恩國一直以來都是世界各地的人們於休假時考慮的旅遊地區之首選。

然而，在這人來人往的國家中，交通運輸便成爲了一個很重要的課題，如何設計、編排及構築通聯道路便成爲了卡恩國中央政府當局的一個大考驗。畢竟，不管你擁有再怎麼樣好的特色，如果沒有便利及順暢的交通網絡的話，恐怕這一切都將只是空談。於是，現在的卡恩國政府當局便決定實施一項前後共三階段的交通改善計劃。其中的第一階段便是先將一些原有的道路拓寬，使得那些道路可以承受更大的車流負載。可是，這麼一來，在某條道路正在施工的那段時間內，我們便暫時地沒有辦法經過該條道路。儘管卡恩國原有的交通網絡已經十分稠密，但這仍會造成一定影響。

現在，你是一間知名旅行社的導遊，平時你最大宗的客戶委託來源就是卡恩國的相關行程導覽活動，因此，熟知卡恩國的你當然知道現在卡恩國正在進行這項交通改善計劃。於是，你決定因應當前的「路況」來修改你原訂的行程。其中，最麻煩的莫過於原本同屬於某一天的兩項行程可能會因爲一條道路的整修而無法在該日同時完成。舉例來說，你原本可能決定好要在結束某個地區 A 的行程後前往某個地區 B ，可是由於某條道路的施工，導致在當天旅遊團無法從地區 A 到地區 B 。爲了避免這種窘境發生，你已經先調查好了第 k 天是哪條道路 r_k 會施工，這麼一來，你應該就可以很快地知道是不是應該修改該天原訂的行程了！

在得到了這些施工的資訊和卡恩國的交通路網圖之後，唯一剩下的就只是根據這些已有的資訊來決定是否需要更動行程了。由於要帶的旅遊團數量太多了，所以，寫一個程式來解決這個問題吧！

爲了簡化我們的問題，我們假設每一天都只有且恰好有兩個行程。也就是對於每一天我們只需要考慮在該天的施工條件下能否能夠從一個地區(前面的行程)到另一個的地區(之後的行程)，並且我們保證這兩個地區將會是**相異**的。

注意！一條道路可能會因爲工程規劃等等的因素導致其需要整修**很多遍**。(不過這些資訊你也都知道了！)

■ 輸入檔說明

第一行有一個整數 T ($T \leq 50$)，代表接下來有幾組測試資料。

每一組測試資料的第一行有兩個數字，N 和 M，代表卡恩國有 N 個相異的地區(分別由正整數 1 編號到 N)，及 M 條**雙向**道路(分別由正整數 1 編號到 M)。($2 \leq N \leq 32000$, $1 \leq M \leq 514000$)

接下來 M 行是卡恩國的交通網路資訊，其中每一行都分別代表了卡恩國中的一條雙向道路，第 i 行包含兩個編號 a_i, b_i，代表該卡恩國的第 i 條道路(即編號爲 i 的道路)連接地區 a_i 和地區 b_i。

需要注意的是，卡恩國裡**不會有**兩條以上、連接同樣一對地區的道路。

之後下一行有一個數字 D ($1 \leq D \leq 201112$)，代表你所要考慮的日子數。接著有 D 行，其中每行都有三個整數 r_k, x_k, y_k，代表在你所關心的第 k 天時編號爲 r_k ($1 \leq r_k \leq M$) 的道路正在整修中，而你想要詢問的是，在該條道路不能使用的情況下是否有方法從編號爲 x_k 的地區到編號爲 y_k 的地區 ($1 \leq x_k, y_k \leq N$)。($1 \leq k \leq D$)

對於一組 x_k, y_k，我們保證在編號爲 r_k 的那條道路可以使用時，**必定**存在一條路徑從編號爲 x_k 的地區到編號爲 y_k 的地區。

■ 輸出檔說明

對於每筆測試資料請輸出恰好一行，該行包含恰好一個有 D 個字元的字串，並滿足以下條件：

如果第 k 天的行程需要被修改(在當天無法從編號爲 x_k 的地區到編號爲

y_k 的地區)，則該輸出字串的第 k 個字元爲 Y ；反之，如果第 k 天的行程不需要被修改，則該輸出字串第 k 個字元爲 N 。

請參考範例輸出的格式。

■ 範例輸入

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

■ 範例輸出

```
YNNYY
Y
NN
```

本頁留白。

題目 F
田忌賽馬

執行時間限制: 5 秒

齊使者如梁，孫臏以刑徒陰見，說齊使。齊使以爲奇，竊載與之齊。齊將田忌善而客待之。忌數與齊諸公子馳逐重射。孫子見其馬足不甚相遠，馬有上、中、下輩。於是孫子謂田忌曰：「君第重射，臣能令君勝。」田忌信然之，與王及諸公子逐射千金。及臨質，臏曰：「今以君之下駟與彼上駟，取君上駟與彼中駟，取君中駟與彼下駟。」既馳三輩畢，而田忌一不勝而再勝，卒得王千金。於是忌進孫子於威王。威王問兵法，遂以爲師。

—『史記。孫子吳起列傳第五』

千年以前，孫臏靠著過人的智謀，巧妙地調整比賽順序，讓三戰皆墨的田忌翻身成兩勝一敗的贏家，也爲自己贏得尊敬和重用。千年以後的今日，賽馬依然是熱門的活動，不過今天你要面對的是更困難的問題。

你和對手各有 N 匹馬，要進行 N 場比賽。一匹馬只限出場一次，同場比賽中速度較快的馬獲勝。若兩匹馬速度一樣，則算平手。

你可以決定你的馬匹的出場順序；而你的對手，就如同齊王，會在第一場比賽出速度最快的馬，第二場出次快的馬，$\cdots$，第 N 場出速度最慢的馬。

除此之外，你還可以決定比賽的時間，全部 N 場比賽都會在你選的這一天進行。在比賽之前，勤勞的你每天都會訓練你的每一匹馬；而你的對手自我感覺非常良好，因此不會訓練他的馬。每一匹馬的素質不同，我們用 a_i 來表示第 i 匹馬的速度，用 b_i 來表示第 i 匹馬的成長率。經過 m 天的訓練，你的第 i 匹馬在第 $m+1$ 天的速度就會是 $a_i + m \cdot b_i$。對手的第 j 匹馬在每一天的速度都是 c_j。

現在你有你和對手共 $2N$ 匹馬的資料，請決定訓練的天數 M，使得在第 $M+1$ 天比賽的時候，你有一個出場順序可以贏得 N 場比賽中的至少 K 場（不包含平手）。

■ 輸入檔說明

第一行有一個整數 T ($T \leq 100$)，代表接下來有幾組測試資料。

每一組測試資料的第一行有兩個數字，N 和 K。

接下來 N 行是你的馬匹的資料，每一行有兩個整數，a_i 和 b_i，代表馬匹的速度和成長率。

再下來 N 行是對手的馬匹的資料，每一行有一個整數 c_j，代表馬匹的速度。（$1 \leq K \leq N \leq 10000$, $0 \leq a_i, c_j \leq 100000000$, $0 \leq b_i \leq 100$）

■ 輸出檔說明

對每筆測試資料輸出一個非負整數 M，代表訓練 M 天後在第 $M+1$ 天舉行賽馬你可以贏得至少 K 場。

如果有不只一個 M 滿足條件，請輸出最小的 M。如果沒有任何 M 滿足條件，請輸出 -1。

■ 範例輸入

```
2
3 2
1 2
2 1
3 2
0
3
4
1 1
1 0
2
```

■ 範例輸出

```
1
-1
```

題目 G
上帝 GOD

執行時間限制: 10 **秒**

各位親愛的子民，這兒是上帝：

最近有一件事情打擾著我。每天我都要從我住的地方到禱告堂聽取人們的禱告，但是我已經一百四十億歲了，這段距離讓我覺得越來越辛苦。天空是由 $M \times N$ 個格子所組成，我住的地方在天空的左上角，禱告堂則在天空的右下角。因爲年齡，我的每一步只能往上、下、左或右其中一個方向，這條路途眞是十分遙遠啊！

幸好，有一群住在空島的人們幫我安裝了一些魔法球，透過這些魔法球讓空間扭曲讓我可以走更短的路。這些魔法球攜帶有各種大小的魔法能量，對某個位置 (x, y) ，如果這個位置的所處的行和列都有相同能量的魔法球，則這個位置就會因爲能量共振產生空間扭曲。例如：如果 $(2, 4)$ 和 $(4, 2)$ 都有能量 2 的魔法球，則在 $(2, 2), (2, 4), (4, 2), (4, 4)$ 這四個格子都會產生空間扭曲。在空間扭曲的格子上，我不只可以往上、下、左或右走還可以往左上、左下、右上或右下走。但是，因爲我是上帝，所以每次利用空間扭曲之後，下次我想再利用另一個空間扭曲的能量，一定要比上次所利用的空間扭曲所攜帶的能量高。

最近我十分地忙碌，每天我都要聽取很多禱告，所以我希望你能幫我計算從我住的地方到禱告堂最少要走幾步路。

■ 輸入檔說明

第一行有一個整數 T $(T \leq 100)$，代表接下來有幾組測試資料。

每一組測試資料的第一行有兩個整數 N, M（$1 \leq N, M \leq 2000$）。第二行有一個數字 K（$0 \leq K \leq 4000$），代表總共有幾個魔法球。接下來有 K 行，每一行有三個數字 x_i, y_i, e_i，其中 (x_i, y_i) 代表第 i 個魔法球的位置與 e_i 代表能量（$1 \leq x_i \leq N, 1 \leq y_i \leq M, 1 \leq e_i \leq 10^9$）。座標從左上角開始算，左上角爲$(1, 1)$。

■ 輸出檔說明

對每筆測試資料輸出一個整數，代表上帝從祂住的地方到禱告堂所需的最短步數。

■ 範例輸入

```
5
3 3
2
1 1 1
2 2 2
3 3
2
1 1 2
2 2 1
3 3
3
1 3 1
3 1 1
2 2 2
4 4
3
2 2 1
2 1 2
3 2 3
14 14
5
4 4 1
9 7 2
5 10 3
7 11 4
12 12 5
```

■ 範例輸出

```
2
3
2
4
22
```

■ 範例輸出說明

範例的第一筆測資最短的走法爲 $(1,1,1) \to (2,2,2) \to (3,3,\times)$，在 $(1,1)$ 用能量 1 的魔法球，之後走到 $(2,2)$ 用 2 的魔法球，之後走到 $(3,3)$。

範例的第二筆測資最短的走法爲 $(1,1,2) \to (2,2,\times) \to (2,3,\times) \to (3,3,\times)$，在 $(1,1)$ 用能量 2 的魔法球，之後走到 $(2,2)$ 但不用（也不能用）魔法球，之後走到 $(2,3)$ 再走到 $(3,3)$。

範例的第三筆測資最短的走法爲 $(1,1,1) \to (2,2,2) \to (3,3,1)$，在 $(1,1)$ 用能量 1 的魔法球，之後走到 $(2,2)$ 用能量 2 的魔法球，之後走到 $(3,3)$。

範例的第四筆測資最短的走法爲 $(1,1,\times) \to (2,1,2) \to (3,2,3) \to (4,3,\times) \to (4,4,\times)$，從 $(1,1)$ 走到 $(2,1)$ 用 2 的魔法球，之後走到 $(3,2)$ 用能量 3 的魔法球，之後走到 $(4,3)$ 再走到 $(4,4)$。

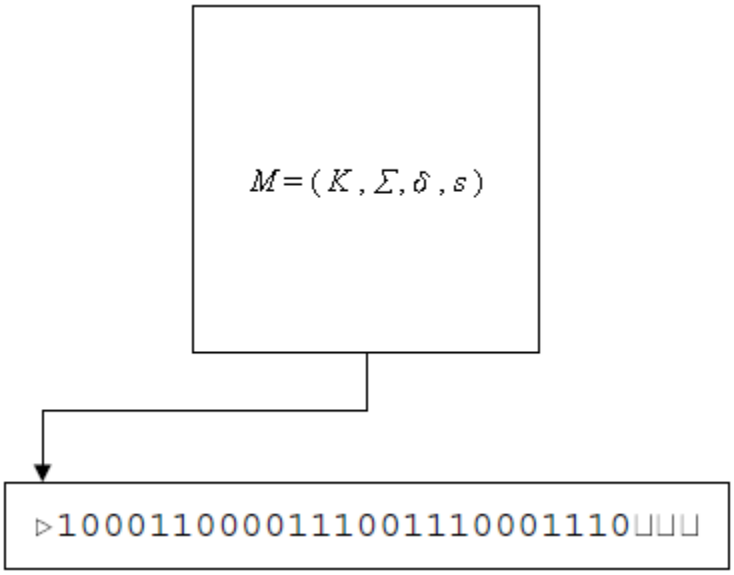
本頁留白。

題目 H
圖靈機

執行時間限制: 10 **秒**

在資訊界中，P 是否就是 NP 是一個有名的未解問題。然而我們可以不正式地定義 P 及 NP ，如果一個問題能讓電腦在多項式時間內解決，那麼這類題目就屬於 P；如果一個題目找到了答案，且可以在多項式時間內驗證這個答案是否正確，則這類的題目就是 NP。然而精準的定義，卻有一套龐大的理論支持著，而這套理論中，最基本的模型就是圖靈機(Turing Machine)。

什麼是圖靈機呢？它的樣子就長得如下，上面的方框框就好像電腦的CPU，他負責運算下一個指令會是什麼。下面有一條帶子存放著字串，這個帶子的長度是無限的，而帶子就好像是記憶體，記錄著現在的運算結果。中間有一個箭頭指到下面的帶子，讓上面的框框能知道目前所指的地方是什麼字母。



讓我們定義圖靈機：一個圖靈機 M 包含著四樣東西，即 $M = (K, \Sigma, \delta, s)$。讓我們一項一項解釋：

- K 爲一個狀態的集合，它定義這個圖靈機有哪些狀態，且一個合理的圖靈機一定有 yes 和 no 這兩種狀態，當圖靈機的狀態切換到 yes，代表所給定的輸入字串是合理的，反之當狀態切到 no，則代表所給定的輸入字串是錯誤的。

- Σ 是一個字母的集合，代表帶子裡可以寫進哪些字母，以上面的例子來講，Σ 有四種字母 $(\triangleright, \sqcup, 0, 1)$，其中 $\triangleright$ 爲開頭字母，$\sqcup$ 爲空字母，這兩個特殊字母也是每個圖靈機必備的。

- δ 是一個函數，它是整個圖靈機的靈魂，負責決定當在什麼狀況會發生什麼事。δ 函數長的像這個樣子：

  δ(現在的狀態, 箭頭指到的字母)
  = (下一個狀態, 在箭頭上指到的地方要被替換成這個字母, 箭頭怎麼走)

  若用一般的話來解釋，就是當圖靈機在某個狀態且箭頭指到某個字母時，它會做三件事情，換成下一個狀態，將箭頭上的字母換掉，將箭頭移動。而箭頭只有三種移法：向左一格，向右一格及維持原地，我們以 $\leftarrow, \rightarrow, -$ 來表示。

- s 爲一開始的狀態，且不會是 yes 或 no。

爲了讓大家了解圖靈機的實際運作，我們以一個能判斷輸入字串長度是否爲偶數的圖靈機當作例子：

- $K = \{yes, no, start, odd, even\}$，也就是說這個圖靈機總共有五種狀態。

- $\Sigma = \{\triangleright, \sqcup, 1\}$，也就是除了圖靈機內設的兩個字母外，這個圖靈機只有額外的一個字母1。

- δ 函數將有 3×3 條 (三種可能的狀態，箭頭可能指到三種字母)：
  - $\delta(start, \triangleright) = (even, \triangleright, \rightarrow)$
  - $\delta(start, \sqcup) = (no, \sqcup, \rightarrow)$
  - $\delta(start, 1) = (no, \sqcup, \rightarrow)$

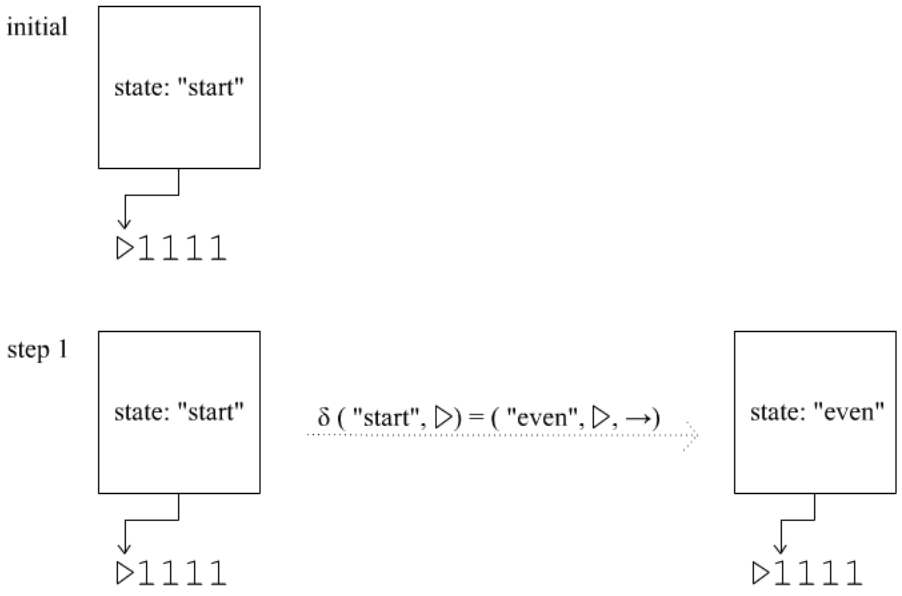
- $\delta(odd, \triangleright) = (no, \sqcup, \rightarrow)$
- $\delta(odd, \sqcup) = (no, \sqcup, \rightarrow)$
- $\delta(odd, 1) = (even, \triangleright, \rightarrow)$
- $\delta(even, \triangleright) = (no, \sqcup, \rightarrow)$
- $\delta(even, \sqcup) = (yes, \sqcup, \rightarrow)$
- $\delta(even, 1) = (odd, \sqcup, \rightarrow)$

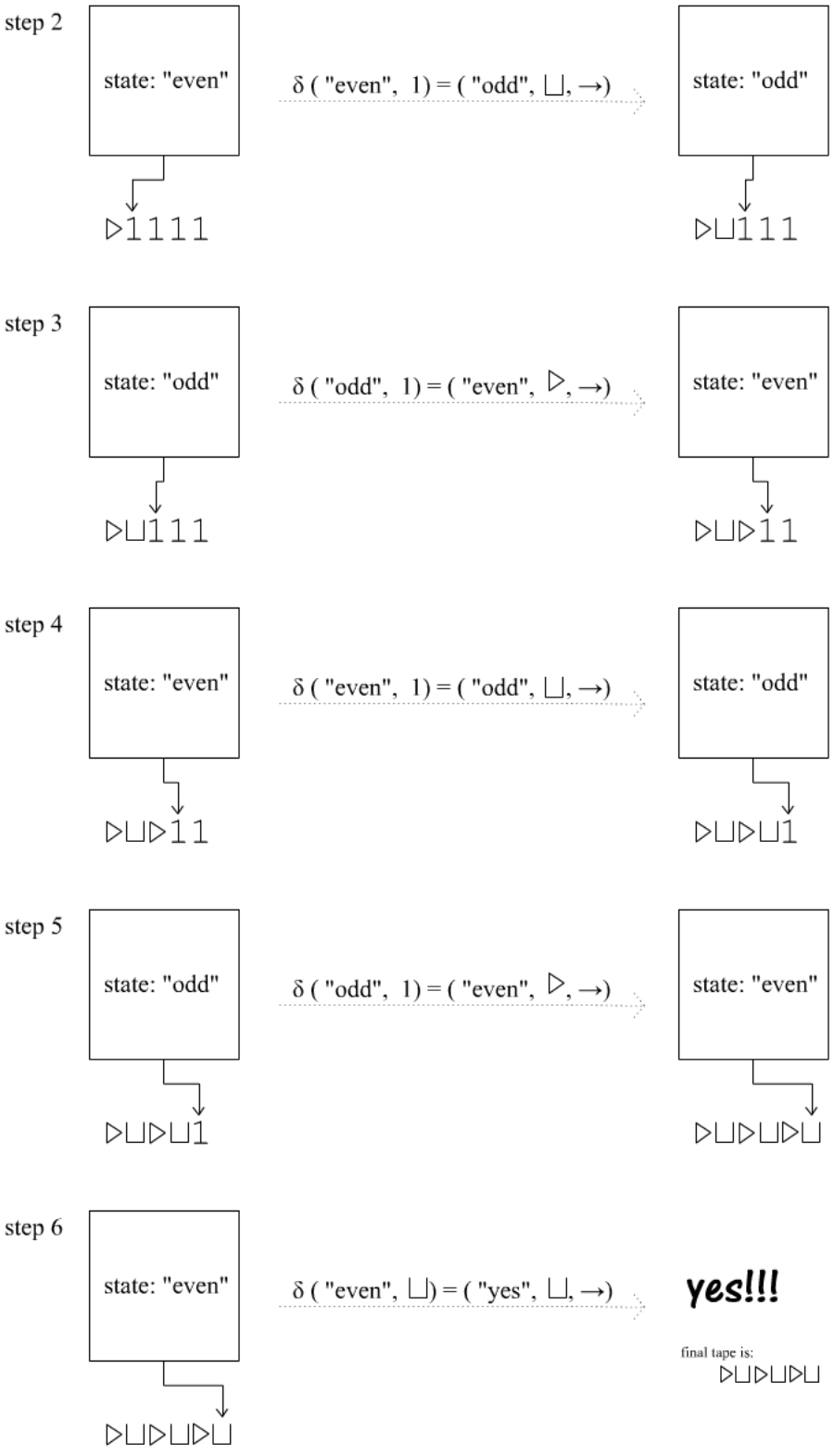
  其中你將會發現第二、三、四、七條是完全不會用到的。而當經過 δ 函數得到狀態爲 yes 或 no 時，其實圖靈機就結束了，所以箭頭取代的字及箭頭移動方向也就不重要。

- s 設爲 $start$。

定義好了圖靈機，接著我們就要將輸入字串丟進這個圖靈機。對於圖靈機的輸入字串，一定要用 Σ 裡面的字母，但不可以使用 $\triangleright, \sqcup$ 這兩項。我們假設輸入字串爲 1111，那麼我們要做的第一件事情，就是將輸入字串丟到圖靈機的帶子裡，做法是這樣：先將第一個字母設成 $\triangleright$，接著放入 1111，因此帶子長度爲 5。

我們將圖靈機的輸入字串放入帶子後，就可以開始執行。




step 2
state: "even"
δ ("even", 1) = ("odd", ␣, →)
state: "odd"
▷1111
▷␣111
step 3
state: "odd"
δ ("odd", 1) = ("even", ▷, →)
state: "even"
▷␣111
▷␣▷11
step 4
state: "even"
δ ("even", 1) = ("odd", ␣, →)
state: "odd"
▷␣▷11
▷␣▷␣1
step 5
state: "odd"
δ ("odd", 1) = ("even", ▷, →)
state: "even"
▷␣▷␣1
▷␣▷␣▷␣
step 6
state: "even"
δ ("even", ␣) = ("yes", ␣, →)
yes!!!
final tape is:
▷␣▷␣▷␣
▷␣▷␣▷␣

你會發現一件事情，在 step 5 的時候，δ 函數指示要往右走，但原本帶子右邊是沒東西的，當這種情況發生時，圖靈機會自動的生出一個 $\sqcup$ 讓長度變成 6，但反之若箭頭往左移，圖靈機的帶子並不會變短。

當圖靈機發現下一個狀態是 yes 或者 no 時，便不會再繼續運作。以上圖的 step 6 作例子，當看到狀態是 yes 時，箭頭上所指的字母不會被替換，且箭頭不會往右移 (這也是爲什麼長度還是 6 的原因)。

而這個圖靈機中，若你給奇數個 1，最後的狀態會是 no。

你的任務就是要模擬不同的圖靈機。

■ 輸入檔說明

第一行有一個整數 T ($T \leq 20$)，代表接下來有幾組測試資料。

每組測試資料代表一個圖靈機，對於每個圖靈機，會有很多行來表示它。第一行有兩個正整數 S, M，其中 S 代表圖靈機裡的 K 有幾個狀態，爲了簡化我們設狀態的名稱爲 1 到 S，且 1 代表 yes，2 代表 no，3 代表 $start$ ，也就是圖靈機裡的 s，因此題目一定保證 $S \geq 3$，同時也保證 $S \leq 100$。而 M 代表圖靈機裡有 $M + 2$ 個字母，分別爲 $\triangleright, \sqcup, a, b, \cdots$，例如 $M = 3$，則有 5 個字母 $\triangleright, \sqcup, a, b, c$。爲了輸入，我們以 s 代替 $\triangleright$，u 代替 $\sqcup$，而 M 最大值爲 13，也就是其它字母只會用到 a 至 m。

接下來的 $(S - 2) \times (M + 2)$ 行，定義著 δ 函式，每一行有五樣東西 P, A, Q, B, D，其中 P 爲一正整數，代表現在的狀態。A 爲一字元，代表箭頭指到的字母。Q 爲一正整數，代表要轉換到的下一個狀態。B 爲一字元，代表箭頭所指到的字母要替換成 B。最後 D 爲一字元，可能爲 $<, >, -$ (小於，大於，減號)，代表箭頭向左移動一格，向右移動一格或不動。

接著爲一正整數 C，代表對這個圖靈機有幾份需要模擬的輸入。最後有 C 行字串，代表要詢問圖靈機的輸入，每行字串最長爲 1000 個字元。請注意，輸入有可能是空字串。測資保證每一組圖靈機間不會有其它空白行。

測資不會讓圖靈機產生無窮迴圈的狀況，每個輸入字串最多只會讓所屬的圖靈機在 100000 步內的模擬保證有結果。測資不會有箭頭在最左邊時，卻還下往左的指令。

■ 輸出檔說明

對於每一個輸入字串，如果結果是 yes，請輸出 `yes xxxxx`，其中 `xxxxx` 是最後帶子上的字串。如果結果是 no，請輸出 `no`。

■ 範例輸入

```
2
5 1
3 s 5 s >
3 u 2 u >
3 a 2 u >
4 s 2 u >
4 u 2 u >
4 a 5 s >
5 s 2 u >
5 u 1 u >
5 a 4 u >
2
aaaa
aaaaaa
5 1
3 s 5 s >
3 u 2 u >
3 a 2 u >
4 s 2 u >
4 u 2 u >
4 a 5 s >
5 s 2 u >
5 u 1 u >
5 a 4 u >
1
aaa
```

■ 範例輸出

```
yes sususu
yes susususu
no
```