2021 網際網路程式設計全國大賽
國中組網路賽

- 本次比賽共 8 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**）。
  輸出和裁判的答案必須完全一致，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 Time Limit Exceeded 的風險。
- 任何題目內提到的「一行」，皆代表以換行字元「`\n`」結尾的字串。
- 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「相對或絕對誤差不超過 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	圈圈叉叉	2 s	1024 MB
B	布林運算式	1 s	1024 MB
C	尋找寶藏	1 s	1024 MB
D	魔法石	2 s	1024 MB
E	火柴棒	1 s	1024 MB
F	一個遊戲	1 s	1024 MB
G	三角撞球	1 s	1024 MB
H	桌遊	1 s	1024 MB

2021 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. 圈圈叉叉

Problem ID: tictactoe

你有玩過圈圈叉叉嗎？圈圈叉叉，又稱井字棋，是一個在大小 3×3 的棋盤上玩的雙人遊戲。兩個玩家一人代表圈、一人代表叉，並且輪流在棋盤空的格子中放上自己的符號，而誰先連成一條線就贏了。

小 Y 是一個天才兒童，所以傳統 3×3 的圈圈叉叉對他來說已經太無趣了。因此，他發明了一種新的版本，並邀請他的好朋友小 P 一起來玩。小 Y 發明的版本是在一個 $N \times N$ 的棋盤上玩，與傳統的規則一樣，一人畫圈、另一人畫叉，並且拿圈的玩家先手。但與傳統的圈圈叉叉規則不一樣的是，小 Y 的遊戲比的不是誰先連成一條線，而是兩人會下到不能再下為止（也就是棋盤已經完全被佔滿），再看誰的連線比較多，一個玩家的得分就是他的連線數量，而很自然的，得分較高的玩家就勝出。

補充說明，在 $N \times N$ 的棋盤上，連線是指有同排、同列、或是同對角線的 N 個相同符號。

小 Y 跟小 P 都覺得新版的圈圈叉叉實在比舊版的好玩太多了，但他們馬上就遇到了一個問題：由於棋盤實在是太大了，遊戲結束後實在無法慢慢的用人力來算分。當然，寫個程式來處理這個問題對於天才兒童小 Y 來說可以說是易如反掌，但可想而知，他還有更重要的事情要做。因此，身為程式競賽參賽者的你們，何不幫幫小 Y 解決這個麻煩的問題呢？

Input

輸入第一行有一個正整數 N，代表圈圈叉叉棋盤的大小。

接著 N 行，每行有一個長度為 N 的字串，代表圈圈叉叉的盤面。保證字串只有 O、X 兩種字元，分別代表圈圈以及叉叉。

- $3 \leq N \leq 2000$

Output

若輸入的盤面不可能為一個合法的結束盤面，輸出 `Impossible`。否則，輸出兩個以空白隔開的整數，分別代表先手玩家的得分以及後手玩家的得分。

Sample Input 1

```
3
000
XXX
0X0
```

Sample Output 1

```
1 1
```

Sample Input 2

```
3
000
000
000
```

Sample Output 2

```
Impossible
```

Sample Input 3

```
4
0XX0
X00X
X00X
0XX0
```

Sample Output 3

```
2 0
```

B. 布林運算式

Problem ID: boolean

最近小 Y 正在學習邏輯，而眾所皆知的，學習邏輯的第一步就是學會「布林變數」。

所謂的「布林變數」，就是一個只能是 0 和 1 的變數，而這些變數可以透過一些專門的「運算子」組成一個「布林運算式」。在本題中，我們只會介紹並使用到「not」、「and」、「or」三種運算子。

一個在本題中合法的「布林運算式」可由下列規則所決定，假設 A、B 是任意兩個合法的布林運算式：

- 任何單一的布林變數是一個合法的布林運算式
- not (A) 是一個合法的布林運算式
- (A) and (B) 是一個合法的布林運算式
- (A) or (B) 是一個合法的布林運算式

在此，所有的「元素」都必須以單一空格隔開，元素包括布林變數、括號以及運算子，一個布林運算式的長度被定義為他的元素個數。

而對於一個合法的布林運算式，他的運算結果如下列規則所決定，假設 A、B 是任意兩個合法的布林運算式：

- 若該運算式為單一的布林變數，則該運算式的結果為**該布林變數被賦予的值**。
- 若該運算式為 not (A)，則若 A 的運算結果是 1，該運算式的運算結果為 0；反之則為 1。
- 若該運算式為 (A) and (B)，則若 A 和 B 的運算結果都是 1，該運算式的運算結果為 1；反之其他狀況則為 0。
- 若該運算式為 (A) or (B)，則若 A 和 B 的運算結果都是 0，該運算式的運算結果為 0；反之其他狀況則為 1。

小 Y 在研究這類布林運算式時發現了下列的等式，假設 A、B 是任意兩個合法的布林運算式，則：

```
( A ) and ( B ) = not ( ( not ( A ) ) or ( not ( B ) ) )
```

這實際上是「笛摩根定理（De Morgan's laws）」的其中一個結論，看到這裡，小 Y 便很好奇，是否有辦法把任意符合本題規則的合法布林運算式，轉換成一個仍舊合法，但**不包含**任何 and 運算子的布林運算式呢？實際上，我們可以證明這總是辦得到的。

注意到，兩個 N 個布林變數的布林運算式若等價，代表對於 2^N 種可能的變數賦值，兩個運算式的運算結果皆相同。

現在小 Y 給你了一個合法的布林運算式，請你給他一個合法的且等價輸入運算式的布林運算式，使得該運算式**不包含**任何 and 運算子。

Input

輸入首行有兩個正整數 N, M，代表布林變數的種類數和輸入運算式的長度。

接下來一行，有一個符合規則且長度為 M 的布林運算式，規則如題敘所述。

- $1 \leq N \leq 10$
- $1 \leq M \leq 1600$
- 所有布林變數將表示成數字，且介於 $1 \sim N$ 之間
- 所有非數字皆為 (,), not, and, or 的其中一種

Output

首行輸出一個介於 $1 \sim 7000$ 之間的數字，代表你轉換過後的布林運算式長度。

接下來一行，輸出一個符合規則並等價輸入運算式、且不包含任何 and 操作的布林運算式。若你輸出的答案符合上述所有規定，則**任何一種答案**皆會被視為 Accepted，否則會被視為 Wrong Answer。

Sample Input 1

```
2 7
( 1 ) and ( 2 )
```

Sample Output 1

```
16
not ( ( not ( 1 ) ) or ( not ( 2 ) ) )
```

Sample Input 2

```
2 13
( not ( 1 ) ) and ( not ( 2 ) )
```

Sample Output 2

```
22
not ( ( not ( not ( 1 ) ) ) or ( not ( not ( 2 ) ) ) )
```

Sample Input 3

```
3 19
not ( ( not ( 3 ) ) and ( ( 2 ) or ( 1 ) ) )
```

Sample Output 3

```
28
not ( not ( ( not ( not ( 3 ) ) ) or ( not ( ( 2 ) or ( 1 ) ) ) ) )
```

Sample Input 4

```
1 7
( 1 ) and ( 1 )
```

Sample Output 4

```
1
1
```

This page is intentionally left blank.

C. 尋找寶藏

Problem ID: treasure

轟龍鞏龍蹦吧咧啦咧，藏滿寶藏的大門終於打開了，就在小 Y 正準備要拿走寶藏的時候，寶藏精靈忽然出了一到謎題，如果小 Y 沒有成功答出來的話，就會被永遠鎖進寶藏箱！

謎題如下：「八八湊七種，九九湊八八種，! @%＃＆＊$＃。」

精靈語言實在太難懂了，翻譯後如下：

已知在寶箱內有 M 個兩兩外型不同的寶藏，他們的重量都是正整數，且和不大於 N。注意到你並不會事先知道 M 的值。

令 $f(i)$ 代表有多少種方法選出若干個寶藏，使得他們的重量和為 i，並且給定一個序列 $a_1, a_2, a_3, \ldots, a_N$ 其中 $a_i = f(i)$ 除以 $10^9 + 7$ 的餘數。

請你找出那 M 個寶藏的重量。注意到即使不同的寶藏的外形不同，重量依然有可能會重複。

Input

輸入的第一行有一個正整數 N。

第二行有 N 個以空格分開的整數 $a_1, a_2, a_3, \ldots, a_N$。

- $1 \leq N \leq 3 \times 10^3$
- $0 \leq a_i < 10^9 + 7$
- 保證至少存在一種合法的寶藏組合滿足條件

Output

輸出第一行只有一個數字 M 滿足 $1 \leq M \leq N$，代表原本的寶箱內有 M 個寶藏。

接下來 M 行，每行有一個正整數 b_i 代表第 i 個寶藏的重量。

如果有多組解，請輸出任意一組就好。

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

D. 魔法石

Problem ID: magicstone

某一天，小 Y 與小 P 來到了 NPSC 國。

在 NPSC 國裡面，小 Y 與小 P 發現，這邊的人們都有非常特別的特徵。因此，小 Y 與小 P 給這邊的人們定義了兩個屬性：Y 屬性與 P 屬性。這兩個屬性都可以用整數來量化。

小 Y 與小 P 還定義，如果兩個人是「類似的」，代表這兩個人的 Y 屬性**或** P 屬性的數值是一樣的。

而在 NPSC 國裡面，有著一個特殊的工具：魔法石。如果一個人使用一個魔法石的話，使用者可以從底下的操作中任選一個執行：

- 讓使用者的 Y 屬性的數值加 1。
- 讓使用者的 Y 屬性的數值減 1。
- 讓使用者的 P 屬性的數值加 1。
- 讓使用者的 P 屬性的數值減 1。

現在，小 Y 與小 P 發現 NPSC 國裡面有 N 個人，第 i 個人一開始的 Y 屬性的數值為 y_i，P 屬性的數值為 p_i。特別的是，在這 N 個人中，所有的 Y 屬性、P 屬性的數值都是偶數。小 Y 與小 P 打算創造出一個新的人，**並且透過發放魔法石給其他 N 個人，讓每個人都跟新創造的人都是「類似的」**。小 Y 與小 P 可以自行決定新創出來的人的 Y 屬性、P 屬性。注意到這兩個數字必須是整數，但是新創的人的 Y 屬性、P 屬性的數值不一定要是偶數。此外，發放的魔法石**不一定要全部使用完畢**，也有可能一顆魔法石都不需要使用。

話說如此，小 Y 與小 P 其實很不願意把過多魔法石交給當地的人民，而在把魔法石給當地居民的時候，還要注意**要給每一個人相同數量的魔法石**，這樣大家才不會起爭執。

於是，小 Y 與小 P 想麻煩你算出，**在給出最少數量的魔法石的情況下，居民們最少可能使用用來修改自身屬性的魔法石總數量是多少**。

Input

輸入的第一行包含一個正整數 T，代表測試資料的數量。

每個測試資料包含 $N+1$ 行。

第一行包含一個正整數 N，代表 NPSC 國人民的數量。

接下來的 N 行，每行包含兩個整數 y_i, p_i，代表第 i 個人的 Y 屬性數值，以及 P 屬性數值。

- $1 \leq T \leq 10^5$
- $1 \leq N \leq 10^5$
- $0 \leq y_i, p_i \leq 10^9$
- y_i, p_i 皆為偶數
- 在這 T 筆測試資料中，N 的總和不會超過 10^5

Output

對於每個測試資料，請輸出兩個整數於一行。第一個整數代表小 Y 與小 P 最少可以給出多少魔法石的數量，第二個整數代表在此前提之下，居民們最少可能使用的魔法石總數量。

Notes

在 Sample Input 1 中，對於第一個測試資料，新創造的人的 Y 屬性、P 屬性可以分別是 $(10, 0)$，這樣原本的那個人就跟新創造的人是「類似的」，因此小 Y 與小 P 不用發放任何魔法石。當然，若新創造的人的 Y 屬性、P 屬性分別是 $(0, 0), (0, 5), (10^9, 0)$ 也都可以達到同樣的效果。

在 Sample Input 1 中，對於第二個測試資料，新創造的人的 Y 屬性、P 屬性可以分別是 $(4, 0)$，這樣原本的兩個人就跟新創造的人是「類似的」，因此小 Y 與小 P 不用發放任何魔法石。當然，若新創造的人的 Y 屬性、P 屬性是 $(0, 4)$ 也可以達到同樣的效果。

在 Sample Input 1 中，對於第三個測試資料，新創造的人的 Y 屬性、P 屬性可以分別是 $(1, 5)$，如此一來，小 Y 與小 P 要發給每個人各一顆魔法石，那四個人可以使用一個魔法石，讓自己的 Y 屬性、P 屬性分別變成 $(1, 0), (1, 2), (4, 5), (6, 5)$，這樣大家就跟新創造的人都是「類似的」。

在 Sample Input 2 中，對於第二個測試資料，新創造的人的 Y 屬性、P 屬性可以分別是 $(6, 6)$，如此一來，小 Y 與小 P 要發給每個人各兩顆魔法石，那五個人可以使用兩個魔法石，讓

自己的 Y 屬性、P 屬性分別變成 $(6,0),(2,6),(2,6),(6,2),(6,10)$，這樣大家就跟新創造的人都是「類似的」。注意到，雖然如果把新創造的人的 Y 屬性、P 屬性設成 $(8,6)$，可以讓總使用的魔法石數量是 8（第一個人用四顆，第二、三個人用兩顆），但是這樣就要發放 $4\times5=20$ 顆魔法石，發放的數量就比 10 還要多。

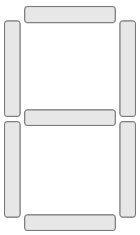
Sample Input 1	**Sample Output 1**
3 1 0 0 2 0 0 4 4 4 0 0 2 2 4 4 6 6	0 0 0 0 4 4

Sample Input 2	Sample Output 2
4 4 2 4 2 2 10 4 4 0 5 4 0 2 4 2 8 8 2 8 10 5 4 8 2 0 6 6 6 0 2 4 5 4 6 0 2 10 0 0 2 0 8	4 2 10 10 10 4 10 6

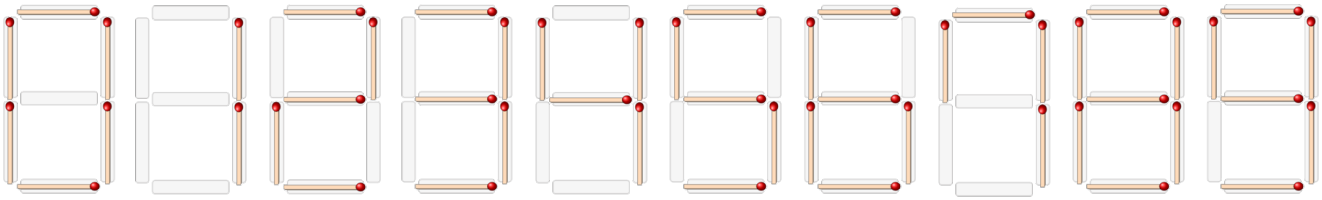
E. 火柴棒

Problem ID: matches

這天，Zisk 收到了不知名人士寄給他的大量火柴棒，身為熱愛資訊科學的天才小孩 Zisk，決定把這些火柴棒以下圖作為框架拼成許多人常見的「數字」們。



Zisk 幫每個 $0 \sim 9$ 之間的數字都定了排列的準則，依序如下圖所示。



準則定好之後，Zisk 便很開心的開始拼湊出各式各樣的數字，但久而久之他便開始感到無趣，聰明的 Zisk 看著他拼好的數字，突然就想到了一個問題：如果他移動**至多**一根火柴棒的話，那他已經拼好的這個數字最大能夠多大呢？

例如，若 Zisk 面前的數字是 369 的話，他可以把 9 的其中一根火柴棒移除，變成 5，並把多出來的火柴棒加到 3 身上，變成 9，整個數字就變成了 965，這也是 Zisk 能拼湊出的最大數字。而若 Zisk 面前的數字是 2 的話，他也可以把 2 左下角的火柴棒往右移變成 3，這樣是這個情況能拼湊出的最大數字。

當然，Zisk 不可以隨意添加火柴棒進來，也不可以在移除一根火柴棒後將他丟棄，一定要放回某個數字上面，並且移動後必須每個數字都個別符合 $0 \sim 9$ 其中一個數字的格式。特別的是，如果移動一根火柴棒之後的數字大於 0 ，那麼這個數字的**首位不得為** 0。

Zisk 知道你也很熱愛資訊科學，因此他決定邀請你決鬥，來較量看看誰可以比較早解出這道題目。

Input

輸入的第一行有一個正整數 T，代表 Zisk 會提出 T 個數字。

接下來 T 行，每行一個非負整數 N，代表 Zisk 這次提出的數字是 N。

- $1 \leq T \leq 2 \times 10^5$
- $0 \leq N < 10^{2 \times 10^5}$
- 若 $N > 0$，則 N 的首位數非 0
- 所有 N 的位數總和不超過 2×10^5

Output

輸出 T 行，每一行分別對應到 Zisk 依序提出的數字 N 在移動**至多**一根火柴棒後，數字最大可以是多少。

Sample Input 1

```
6
369
10429
111111111
638472930237487013794817923314134313 41
2
151515151515151515
```

Sample Output 1

```
965
19429
111111111
938472930237487013794817923314134313 41
3
151515151515151515
```

F. 一個遊戲

Problem ID: game

某天，小 Y 告訴小 P 一個遊戲，那個遊戲叫做一個遊戲。

一個遊戲是一個遊戲。一個遊戲是一個一個人玩的遊戲。一個遊戲的玩家必須從 1 開始數到 N，但必須要跳過一些小 Y 不喜歡的數字。

對於一個正整數 x，以及一個小 Y 會告訴你的正整數 K（K 介於 1 到 9 之間），只要滿足下列至少一個條件，那麼小 Y 就會不喜歡 x 這個數字：

- x 是 K 的倍數。
- x 寫成十進位之後包含 K 這個數字。

否則，小 Y 就會喜歡 x 這個數字。

舉例來說，如果 $K = 3$，那麼 $2, 8, 14$ 都是小 Y 喜歡的數字，而 $3, 12, 23$ 都是小 Y 不喜歡的數字。

小 P 現在想要開始玩一個遊戲。給定 N 以及 K，請你告訴小 P，在一個遊戲的過程中，他會跳過多少小 Y 不喜歡的數字。

Input

輸入只有一行，包含兩個以空格分開的正整數，分別是 N 及 K。

- $1 \leq N \leq 5 \times 10^5$
- $1 \leq K \leq 9$

Output

輸出一個整數，代表在一個遊戲中，小 P 會跳過多少數字。

Sample Input 1	Sample Output 1
7 3	2

Sample Input 2	Sample Output 2
100 9	27

G. 三角撞球

Problem ID: triangle

小 Y 研發了一種新遊戲叫做三角撞球，三角撞球是一種在邊長為 N 的正三角形球檯上進行的遊戲。下圖為 $N = 2$ 的正三角形球檯：

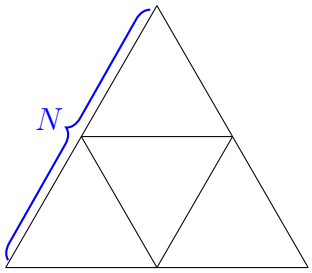


Figure G.1: $N = 2$.

小 P 身為小 Y 的朋友，他對這個球檯很有興趣，他想要知道從距離正上方頂點 K 的位位置水平地將球射入，需要經過幾次反彈才會回到原位，下圖為 $N = 4, K = 1$ 的情形，一共需要反彈 6 次。

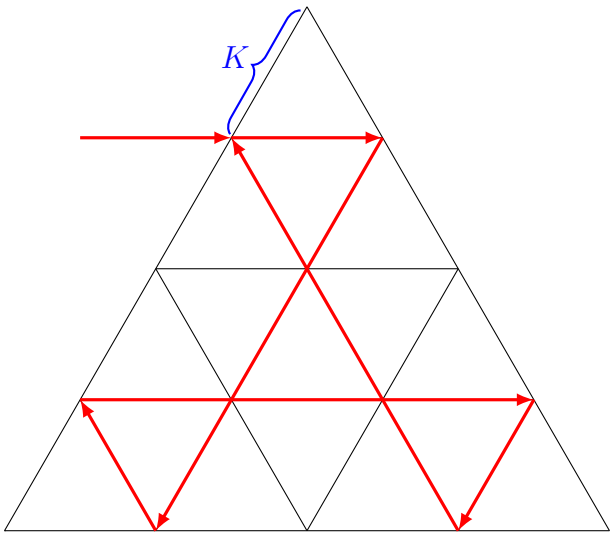


Figure G.2: $N = 4, K = 1$.

Input

輸入僅包含一行共兩個正整數 N, K，分別代表撞球檯的大小以及球進入的位置。

- $2 \leq N \leq 10^3$
- $0 < K < N$

Output

請輸出一個正整數代表球反彈的次數。

Sample Input 1	Sample Output 1
4 1	6

Sample Input 2	Sample Output 2
2 1	3

Sample Input 3	Sample Output 3
3 1	6

Sample Input 4	Sample Output 4
4 2	3

H. 桌遊

Problem ID: boardgame

某天，小 Y 在桌上看到了小 P 前陣子在玩的桌遊。

這個桌遊有一個很長的盤面，以及一些小塊的方塊，用來放上這個盤面。

每一個方塊上面會寫著一個符號以及一個數字，符號是加號 + 或是乘號 * 的其中一個，而數字是介於 1 到 10^9 的整數。例如，一個方塊可能是 +87 或者是 *7122。

所謂很長的盤面，指的是可以依序將每個方塊放上這個盤面。這個盤面恰巧可以放上所有的方塊，而為了結束這個遊戲，玩家也必須將所有的方塊放上這個盤面。

當放完所有方塊之後就是計分環節。

首先，玩家一開始的分數是 0 分。接下來，按照玩家放上方塊的順序，去改動玩家的分數。例如，對於 +3，+5，*7 這三個方塊，如果玩家放上的方塊的順序是 +3，*7，+5，那麼玩家最後的分數就會是 $(((0+3)*7)+5)=26$；但如果玩家放上方塊的順序是 +5，*7，+3，那麼玩家最後的分數就會是 $(((0+5)*7)+3)=38$。

現在，知道了這個遊戲內容的小 Y 不禁覺得這樣的遊戲好像太無聊了點。於是，他決定把每個方塊加上一個限制：對於第 i 個方塊，只有在盤面上的方塊不少於 c_i 個的時候才能放上盤面。

小 Y 不禁開始思考，在這樣子的條件之下，最高可以獲得的得分是多少？

Input

輸入的第一行是一個正整數 N，代表方塊的總數。

接下來的 N 行，第 i 行會有一個字串 s_i 跟一個數字 c_i。s_i 是由兩個部份所構成：第一個字元會是 + 或 * 的其中一個，然後後面接著一個正整數 x_i。這代表第 i 個方塊上面寫的字，以及放上這個方塊之前，版面上最少要有多少個方塊。

- $1 \le N \le 2 \times 10^5$
- $1 \le x_i \le 10^9$
- $0 \le c_i \le N-1$

Output

如果在這樣的條件下無法放上所有的方塊，請輸出 -1。

否則，請輸出最高可以獲得的分數。因為這個分數可能會很大，所以請輸出這個分數除以 $10^9 + 7$ 的餘數。

請注意，要輸出的數字是最高的分數除以 $10^9 + 7$ 之後的餘數，並不是分數除以 $10^9 + 7$ 之後餘數的最高數值。

舉例來說，如果最高分的策略可以得到 $10^9 + 8$ 分，且有另一種可以得到 $10^9 + 6$ 的策略，雖然 $10^9 + 8$ 除以 $10^9 + 7$ 後的餘數為 1，你仍然應該輸出 1 而不是 $10^9 + 6$。

Sample Input 1	Sample Output 1
3 +3 0 *5 0 +7 0	50

Sample Input 2	Sample Output 2
3 +3 0 *5 2 +7 2	-1