囚犯的挑戰 (Prisoner Challenge)

在一個監獄裏有500名囚犯。 一天，監獄長為他們提供了一個解脫的機會。他在一個房間裡放了兩個裝有錢的袋子，袋子 A 和袋子 B。每個袋子內有1 至 N 個硬幣（含）。袋子A內的硬幣數量和袋子B內的硬幣數量是**不同的**。監獄長向囚犯們提出挑戰。囚犯們的目標就是要識別出較少硬幣數量的那個袋子。

房間裡除了錢袋之外，還有一塊白板。在白板上任何時候都會被寫上一個數字。最初，白板上的數字是 0。

然後，監獄長讓囚犯一個一個進入房間。進入房間的囚犯並不知道在他們進入房間之前有誰或有多少個囚犯進去過。每個囚犯進入房間後，他們都會閱讀當前寫在白板上的數字。讀完數字後，他們必須選擇袋子 A 或袋子 B。然後囚犯可以**檢查**他所選擇的袋子，從而知道這個袋子裡面的硬幣數量。然後，囚犯必須執行以下兩個**動作**其中之一：

- 用一個非負的整數覆蓋白板上的數字，然後離開房間。 請注意，他們可以更改或保留當前的數字。之後挑戰繼續進行（直至全部 500 名囚犯都已經進入房間）。
- 指出其中一個袋子為硬幣數量較少的袋子。這個挑戰立即結束。

注意，監獄長不會要求已經離開房間的囚犯再次進入房間。

如果其中一個囚犯能正確識別出硬幣數量較少的袋子，囚犯們就會贏得挑戰。但是如果他們當中任何一個囚犯錯誤地識別了袋子，或者全部 500 個囚犯都進入了房間並且沒有嘗試去識別較少的硬幣數量的袋子，他們就輸了。

在挑戰開始前，囚犯們都聚集在監獄大廳，他們決定將挑戰的**策略**分成三步。

- 他們選擇一個非負整數 x，這是他們可能想在白板上寫上的數字的最大值。
- 他們決定，無論寫在白板上數字 i $(0 \leq i \leq x)$為何值，在進入房間並從白板上讀取數字 i 時，囚犯就應該知道要檢查哪個袋子。
- 他們在知道所選袋子裡的硬幣數量後，房間裡的囚犯就應該要決定採取什麼行動。具體來說，在看到寫在白板上的數字 i $(0 \leq i \leq x)$ 和檢查所選袋子中看到的的硬幣數量 j $(1 \leq j \leq N)$後，囚犯就要作出以下其中一個決定
  - 在白板上寫上0 和 x （含）之間的一個數字，或者
  - 哪一個袋子應該被識別為硬幣數量較少的袋子。

在贏得挑戰後，監獄長會在囚犯們服刑 x 天後全部釋放。

您的任務是為囚犯設計一個策略，以確保他們贏得挑戰（無論袋子A和袋子B中有多少硬幣）。您的解決方案的分數取決於 x 的值（有關的詳細信息，請參閱子任務部分）。

編程細節

你應該編寫以下的子程序：

```
int[][] devise_strategy(int N)
```

- N: 每個袋子中最大可能的硬幣數量。
- 這個子程序應該返回一個由 $N+1$ 個組成的整數數組所組成的數組 s，代表你的策略。 數組 s 的長度是x 的值減1。 對於每個 i 使得 $0 \leq i \leq x$，數組 $s[i]$ 表示如果囚犯在進入房間時從白板上讀取的數字 i後應該要做什麼：
    1. 如果囚犯應該去檢查袋子 A，$s[i][0]$ 的值為 0，如果囚犯應該去檢查包 B，則 $s[i][0]$ 的價值是 1。
    2. 設 j 為所選袋子中看到的硬幣數量。然後囚犯應執行以下操作：
        - 如果 $s[i][j]$ 的值為 -1，犯人應將袋子 A 識別為硬幣數量較少的袋子。
        - 如果 $s[i][j]$ 的值為 -2，犯人應將袋子 B 識別為硬幣數量較少的袋子。
        - 如果 $s[i][j]$ 的值是一個非負數，犯人應該把這個數寫在白板上。注意 $s[i][j]$ 的值最大只能是 x。
- 這個子程序只會被調用一次。

範例

考慮以下的調用：

```
devise_strategy(3)
```

令 v 表示囚犯進入房間時從白板上讀取的數字。 正確的策略以下其中之一：

- 如果 $v = 0$（包括初始編號），檢查袋子 A。
    - 如果它包含 1 個硬幣，將袋子 A 識別為硬幣數量較少的袋子。
    - 如果它包含 3 個硬幣，將袋子 B 識別為硬幣數量較少的袋子。
    - 如果它包含 2 個硬幣，請在白板上寫下 1（覆蓋 0）。
- 如果 $v = 1$，檢查袋子 B。
    - 如果它包含 1 個硬幣，將袋子 B 識別為硬幣數量較少的袋子。
    - 如果它包含 3 個硬幣，將袋子 A 識別為硬幣數量較少的袋子。
    - 如果它包含 2 個硬幣，請在白板上寫 0 （覆蓋 1）。 請注意，這種情況永遠不會發生，因為我們可以得出結論，兩個袋子都包含 2 個硬幣，這是不允許的。

要報告此策略，該子程序應返回 `[[0, -1, 1, -2], [1, -2, 0, -1]]`。 返回數組的長度為 2，因此對於此返回值，x 的值為 $2 - 1 = 1$。

限制條件

- $2 \leq N \leq 5000$

子任務

1. (5 分) $N \leq 500, x$ 的值不能大於 500。
2. (5 分) $N \leq 500, x$ 的值不能大於 70。
3. (90 分) x 的值不能大於 60。

在任何的測試用例中，如果`devise_strategy`返回的數組不是正確的策略，你的子任務分數將會為 0 分。

在子任務 3 中，您可以獲得部分分數。 令 m 為該子任務中所有測試樣例返回的數組 x 的最大值。 根據下表計算您對該子任務的分數：

條件	分數
$40 \leq m \leq 60$	20
$26 \leq m \leq 39$	$25 + 1.5 \times (40 - m)$
$m = 25$	50
$m = 24$	55
$m = 23$	62
$m = 22$	70
$m = 21$	80
$m \leq 20$	90

樣例測試程式

樣例測試程式會讀取以下形式的輸入：

- 第 1 行: N
- 第 $2 + k$ $(0 \leq k)$行: $A[k]$ $B[k]$
- 最後1行: -1

除了第一行和最後一行之外的每一行都代表一個場景。 我們將第 $2 + k$ 行中描述的場景稱為場景 k。 在場景 k 中袋子 A 中有 $A[k]$ 個硬幣，袋子 B 中有 $B[k]$ 個硬幣。

樣例測試程式首先會調用`devise_strategy(N)`。 x 的值是調用後返回的數組的長度減1。 然後，如果樣例測試程式檢測到 `devise_strategy` 返回的數組不符合實現細節中描述的限制條件，它會打印以下錯誤消息之一並退出：

- `s is an empty array`: s 是一個空的數組（不代表一個有效的策略）
- `s[i] contains incorrect length`: 存在一個索引值 i $(0 \leq i \leq x)$ 使得數組 $s[i]$ 的長度不是 $N + 1$.

- `First element of s[i] is non-binary`: 存在一個索引值 i ($0 \leq i \leq x$) 使得 $s[i][0]$ 的值既不是 0 也不是 1.
- `s[i][j] contains incorrect value`: 存在一個值 i, j ($0 \leq i \leq x, 1 \leq j \leq N$) 使得 $s[i][j]$ 的值不是 -2 至 x之間。

否則，樣例評分程式會產生兩個輸出。

首先，樣例評分程式以以下格式打印您的策略輸出：

- 第 $1 + k$ ($0 \leq k$)行: 在你的策略中場景k的輸出。 如果應用該策略導致囚犯將袋子 A 識別為硬幣較少的依子，則輸出為字母 A。 如果應用該策略導致囚犯將袋子 B 識別為硬幣較少的袋子，則輸出為字母B。 如果應用該策略不會導致任何囚犯識別出硬幣較少的袋子，則輸出為字母X。

其次，樣例評分程式在當前目錄中寫入一個文件 `log.txt`，格式如下：

- 第 $1 + k$ ($0 \leq k$)行: $w[k][0]$ $w[k][1]$ ...

第 $1 + k$ 行的順序對應於場景 k，並描述了寫在白板上的數字。 具體來說，$w[k][l]$ 是 第$(l + 1)^{th}$ 個囚犯進入房間時白板上的數字。