香港電腦奧林匹克競賽 2025/26 初級組

題目總覽

編號	名稱	執行時間限制	記憶體限制	子任務
J261	立方體日期	1.000 s	256 MB	12 + 19 + 14 + 27 + 21 + 7
J262	保齡得分還原	1.000 s	256 MB	10 + 13 + 14 + 17 + 18 + 28
J263	抽檢除錯	1.500 s	256 MB	10 + 8 + 13 + 19 + 22 + 28
J264	害羞的學生	1.000 s	256 MB	9 + 8 + 17 + 19 + 20 + 27

注意:
除非特別注明，否則輸入輸出將依照以下格式:

- 同一行中，數字與數字或字元之間需有一個空格。
- 同一行中，字元與字元之間並無空格。
- 每個字串需放在獨立的行。
- 輸出將自動被修正如下：每行最尾的連續的空格會被刪除，及在輸出最後補上換行符(如沒有)。其他格式問題則不會修正。

C++ 使用者請注意 `cin` / `cout` 可能導致輸入輸出樽頸使程式執行變慢。

有些題目可能需要使用 64 位元整數。在 C++ 中它是 `long long` 而其 `scanf`/`printf` 代號是 `%lld`。

所有題目均有細分多個子任務，你需要通過該子任務中的所有測試數據才能得到分數。

J261 - 立方體日期

執行時間限制: 1.000 s / 記憶體限制: 256 MB

愛麗絲喜歡立方體。今年，她收到了 4 個立方體作為生日禮物！每個立方體的 6 個面上都寫上一個相同的數位。

愛麗絲用 D_1、D_2、D_3 和 D_4 來表示 4 個立方體上寫上的數位。對於 $1 \le i \le 4$，D_i 的值滿足 $0 \le D_i \le 9$。

隨著 2026 年的來臨，愛麗絲希望使用全部 4 個立方體來顯示 2026 年的一個日期。她必須使用剛好 2 個立方體來顯示月份，和另外 2 個立方體來顯示日子。顯示的日期必須是合法的，亦即是月份必須為 1 和 12 之間（含），日子必須至少為 1 且不超過該月份的天數。2026 年每個月份的天數如下表所示：

月份	1	2	3	4	5	6	7	8	9	10	11	12
天數	31	28	31	30	31	30	31	31	30	31	30	31

例如，當 4 個立方體的寫上的數位分別為 1、1、2 和 3，愛麗絲可以顯示日期 `12-31`，即十二月三十一日。但是，當寫上的數位分別為 1、9、9 和 9，愛麗絲則沒有可以顯示的合法日期。

給定數位 D_1、D_2、D_3 和 D_4，請幫助愛麗絲顯示一個 2026 年的合法日期，或者判斷無法顯示一個合法日期。注意，立方體不可以被旋轉或翻轉，例如，不可以使用寫上數位 6 的立方體來顯示數位 9，反之亦然。

輸入

輸入包含 4 行。第 i 行含有一個數位 D_i，第 i 個立方體寫上的數位。

保證立方體上寫上的數位以 **非遞減順序** 給出，即 $D_1 \le D_2 \le D_3 \le D_4$。

輸出

如果無法使用數位 D_1、D_2、D_3 和 D_4 顯示一個合法日期，輸出 `No`。

否則，在第一行輸出 `Yes`。在第二行輸出一個格式為 `MM-DD` 的日期，其中 `MM` 和 `DD` 為兩位數（可包含前導零），代表日期的月份和日子。

如果有多個可以顯示的合法日期，你可以輸出任意一個。

輸出格式

輸出 **分大小寫**，例如，輸出 `no` 和 `NO` 會被視為不正確。

樣例

1

輸入	輸出
1 1 2 3	Yes 12-31

其他可以接受的答案包括 `11-23` 和 `12-13`。但是，`11-32` 和 `31-12` 皆不可接受。

	輸入	輸出
2	0 2 2 9	Yes 09-22

注意，日期 `02-29` 在 2026 年並 **不** 存在。

3	0 0 0 0	No
4	1 9 9 9	No
5	0 0 1 1	Yes 01-01

子任務

對於所有數據:
$0 \le D_1 \le D_2 \le D_3 \le D_4 \le 9$

	佔分	約束條件
1	12	$D_1 = D_2 = 0$ $D_3 \neq 0$ $D_4 \neq 0$
2	19	$D_1 = D_2 = 0$
3	14	$D_1 = 0$ $D_2 = 1$ $D_3 \geq 1$ $D_4 \geq 1$
4	27	$D_1 = 0$
5	21	$D_1 = 1$ $D_2 \geq 1$ $D_3 \geq 1$ $D_4 \geq 1$
6	7	無額外約束

J262 - 保齡得分還原

執行時間限制: 1.000 s / 記憶體限制: 256 MB

愛麗絲正在訓練，目標成為職業保齡球選手！她打算加入位元組國的保齡球聯賽，該聯盟採用以下規則：

比賽由任意多個 計分格 組成。每一個計分格有 10 支保齡球瓶，愛麗絲需要用最多兩顆保齡球來擊倒它們。玩家在該計分格得到的分數會依照結果而不同，如下所示：

- 全中：在該計分格中，第一顆球就擊倒全部 10 支瓶。玩家可得 30 分，並且下一個計分格會隨即開始。
- 補中：在該計分格中，兩顆球合共擊倒全部 10 支瓶，且第一顆球沒有擊倒全部瓶子。玩家可得 $10 + x$ 分，其中 x 為該計分格第一顆球擊倒的瓶數。
- 未補中：在該計分格中，兩顆球都投完後仍有瓶子留下。玩家可得 y 分，其中 y 為該計分格兩顆球擊倒的瓶子總數。

例如，若愛麗絲第一顆球擊倒全部 10 支瓶，則這一計分格是全中，她可得 30 分。若愛麗絲第一顆球擊倒 3 支瓶，第二顆球擊倒 7 支，則這一計分格是補中，她可得 10 + 3 = 13 分。

然而，保齡球館的記分系統發生故障了！系統只記錄了 N 次**擊倒至少一支瓶子**的投球瓶數。由於愛麗絲已經練習了很久而且很疲倦，她請你計算她可能得到的最大分數。

輸入

輸入的第一行包含一個整數 N，為系統所記錄的保齡球投球次數。

輸入接下來的 N 行各包含一個整數 A_i，第 i 次被記錄投球所擊倒的瓶數。

輸出

輸出為一個整數，表示愛麗絲可能得到的最大分數。

樣例

	輸入	輸出
1	4 10 3 6 4	49

其中一種可能的擊倒瓶數的序列為 $(10, 3, 0, 6, 4)$。注意擊倒了 0 球的保齡球沒有被系統紀錄。
在這種情況下，愛麗絲會得到的分數如下：

- 第一顆球算作全中，愛麗絲可得 30 分。
- 接下來兩顆球算作未補中，愛麗絲可得 3 + 0 = 3 分。
- 最後兩顆球算作補中，愛麗絲可得 10 + 6 = 16 分。

因此，愛麗絲總共可以得到 30 + 3 + 16 = 49 分。
可以證明這是愛麗絲可能得到的最大分數。

	輸入	輸出
2	4 7 3 6 7	30

其中一種可以達到最大分數的擊倒瓶數序列為 (7, 3, 6, 0, 7, 0)。
注意 (7, 3, 6, 7) 並不是一個有效的序列，因為第二局會有 6 + 7 = 13 支瓶被擊倒，超過一局最多可以擊倒的 10 支瓶。

3	5 8 2 8 2 8	44

其中一種可以達到最大分數的擊倒瓶數序列為 (8, 2, 8, 2, 8, 0)。

子任務

對於所有數據:
$1 \le N \le 10^5$
對於 $1 \le i \le N$, $1 \le A_i \le 10$

	佔分	約束條件
1	10	$N = 2$ $A_1 \ne 10$ $A_2 \ne 10$
2	13	對於 $1 \le i \le N$，$A_i < 5$
3	14	對於 $1 \le i \le N$，$A_i + A_{i+1} \ne 10$
4	17	$N \ge 2$ 對於 $1 \le i < N$，$A_i + A_{i+1} = 10$ $A_1 > A_2$
5	18	對於 $1 \le i < N$，$A_i + A_{i+1} = 10$
6	28	無額外約束

J263 - 抽檢除錯

執行時間限制: 1.500 s / 記憶體限制: 256 MB

鮑伯在香城科技公司工作。該公司有一個舊代碼庫，共有 L 行代碼，以 1 至 L 編號。

代碼包含 N 個互不重疊的有錯誤部分。第 i 個部分覆蓋從第 U_i 行到第 V_i 行（含）的連續範圍。最初，這些範圍內每一行都有錯誤，所有其他行則沒有錯誤。對於所有 $1 \le i \le N$，這些部分滿足 $1 \le U_i \le V_i \le L$，且它們已排序，及於每兩個錯誤部分之間有至少一行無誤的代碼，即對於所有 $1 \le i < N$，$V_i + 1 < U_{i+1}$。

鮑伯從第 1 天開始每天執行一套固定工序。每一天，鮑伯有一個起始行號 t。他按順序檢查第 $t, t+S, t+2S, \ldots$ 行，其中 S 是一個固定的步距。當他檢查完行 X 為錯誤時，他會立即停止檢查。鮑伯很聰明，所以他總能判斷一行是否有錯誤。如果他檢查超越第 L 行之後仍未找到任何錯誤，他在當天不修復任何代碼。

找到第一個有錯誤的行 X 後，鮑伯進入除錯模式。他修復第 X 行（使該行變為無誤），然後修復第 $X-1$ 行，然後是第 $X-2$ 行，依此類推。他繼續按遞減的順序修復錯誤的行數，直到遇到一行無錯誤的代碼或修復完第 1 行為止，然後他停止當天的工作。請注意，一旦某行被修復，它在所有後續日子中都保持無誤。

例如，假設步距為 $S = 3$，起始行號為 $t = 3$，最初第 4、6-11 和 14-15 行均有錯誤。鮑伯按順序檢查第 $3, 6, 9, 12, 15, \ldots$ 行。第 6 行是他遇到有錯誤的第一行，所以 $X = 6$。

第 1-3 行			第 4-6 行			第 7-9 行			第 10-12 行			第 13-15 行		
0	0	0	1	0	1	1	1	1	1	1	0	0	1	1

底線/藍色的單元格顯示鮑伯檢查的行。

鮑伯接着修復第 6 行，但立即停止，因為第 5 行已經沒有錯誤。因此，鮑伯在這一天修復了 1 行。

第 1-3 行			第 4-6 行			第 7-9 行			第 10-12 行			第 13-15 行		
0	0	0	1	0	0	1	1	1	1	1	0	0	1	1

粗體/綠色的單元格顯示鮑伯修復的行。

鮑伯使用一個循環時間表來決定每一天的起始行號。他有一個包含 P 個起始行號的序列 $T_1, T_2, \ldots, T_P$（其中 $1 \le T_i \le S$）。在首 P 天，鮑伯分別使用行號 $T_1, T_2, \ldots, T_P$。在第 $P+1$ 天，時間表開始循環，所以鮑伯再次使用 T_1，依此類推。正式來說，在第 D 天，鮑伯使用的起始行號為 T_k，其中 $k = ((D-1) \bmod P) + 1$。

繼續上面的例子，假設鮑伯有行號序列 $T_1 = 3, T_2 = 1$（所以 $P = 2$）。這意味著鮑伯在第 1、3、5 天等的起始行號為 $T_1 = 3$，及在第 2、4、6 天等的起始行號為 $T_2 = 1$。在第 1 天之後（鮑伯如上所述修復了第 6 行），第 4、7-11 和 14-15 行仍有錯誤。在第 2 天，鮑伯使用起始行號 $T_2 = 1$。他按順序檢查第 $1, 4, 7, 10, 13, \ldots$ 行。第 4 行是有錯誤的第一行。鮑伯修復第 4 行，但立即停止，因為第 3 行沒有錯誤。因此，鮑伯在當天修復了 1 行。在第 3 天，時間表開始循環，所以鮑伯再次使用行號 $T_1 = 3$。他檢查第 $3, 6, 9, 12, 15, \ldots$ 行。第 9 行是有錯誤的第一行。鮑伯然後修復第 9、8、7 行，總共修復 3 行，依此類推。

下表為此例子從第 2 天開始的完整修復過程。

第 2 天：起始行號 $T_2 = 1$，修復 1 行

步驟	第 1-3 行			第 4-6 行			第 7-9 行			第 10-12 行			第 13-15 行		
檢查後	0	0	0	1	0	0	1	1	1	1	1	0	0	1	1
修復後	0	0	0	0	0	0	1	1	1	1	1	0	0	1	1

第 3 天：起始行號 $T_1 = 3$，修復 3 行

步驟	第 1-3 行			第 4-6 行			第 7-9 行			第 10-12 行			第 13-15 行		
檢查後	0	0	0	0	0	0	1	1	1	1	1	0	0	1	1
修復後	0	0	0	0	0	0	**0**	**0**	**0**	1	1	0	0	1	1

第 4 天：起始行號 $T_2 = 1$，修復 1 行

步驟	第 1-3 行			第 4-6 行			第 7-9 行			第 10-12 行			第 13-15 行		
檢查後	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1
修復後	0	0	0	0	0	0	0	0	0	**0**	1	0	0	1	1

第 5 天：起始行號 $T_1 = 3$，修復 2 行

步驟	第 1-3 行			第 4-6 行			第 7-9 行			第 10-12 行			第 13-15 行		
檢查後	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1
修復後	0	0	0	0	0	0	0	0	0	0	1	0	0	**0**	**0**

第 6 天：起始行號 $T_2 = 1$，修復 0 行

步驟	第 1-3 行			第 4-6 行			第 7-9 行			第 10-12 行			第 13-15 行		
檢查後	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
修復後	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

從第 6 天開始鮑伯不再修復任何代碼。第 11 行的錯誤將永遠存在，因為鮑伯的檢查模式（起始行號為 3 和 1，步距為 3）使鮑伯永遠不會檢查第 11 行。

你需要回答 Q 個查詢。第 i 個查詢（$1 \le i \le Q$）給出一天 D_i。你必須計算鮑伯在該特定日子中 **修復的行數**。請注意，即使存在沒有被查詢的日子， 代碼的錯誤狀態仍會從第 1 天開始持續存在並隨時間演變。

輸入

第一行包含兩個整數 N 和 L。

接下來 N 行描述最初的有錯誤行數。第 i 行（$1 \le i \le N$）包含兩個整數 U_i 和 V_i。

接下來一行包含兩個整數 S 和 P。

接下來一行包含 P 個整數 $T_1, T_2, \ldots, T_P$。

接下來一行包含一個整數 Q。

接下來一行包含 Q 個整數 $D_1, D_2, \ldots, D_Q$。

輸出

輸出一行包含 Q 個以空格分隔的整數。第 j 個整數是鮑伯在第 D_j 天修復的行數。

樣例

1

輸入

```
3 15
4 4
6 11
14 15
3 2
3 1
5
1 3 5 6 100
```

輸出

```
1 3 2 0 0
```

參見題目。

2

輸入

```
3 25
5 9
12 15
20 23
6 3
2 3 2
6
1 2 3 5 8 10
```

輸出

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

鮑伯在第 1 天修復了第 5 至 8 行。
鮑伯在第 2 天修復了第 9 行。
鮑伯在第 3 天修復了第 12 至 14 行。
鮑伯在第 4 天修復了第 20 行。
鮑伯在第 5 天修復了第 15 行。
鮑伯在第 6 天和第 7 天沒有修復任何代碼。
鮑伯在第 8 天修復了第 21 行。
從第 9 天開始，鮑伯沒有修復任何代碼。

3

輸入

```
4 67
7 26
37 37
40 42
56 66
10 4
6 7 10 7
11
1 2 3 4 5 6 7 8 9 10 11
```

輸出

```
10 1 3 1 6 2 1 0 9 0 0
```

4

輸入

```
1 24
1 19
5 3
2 1 5
10
1 2 3 4 5 6 7 8 9 10
```

輸出

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

子任務

對於所有數據:
$1 \le N \le 1500$
$1 \le L \le 10^9$
對於 $1 \le i \le N$，$1 \le U_i \le V_i \le L$
對於 $1 \le i \le N-1$，$V_i + 1 < U_{i+1}$
$1 \le S \le L$
$1 \le P \le 1500$
對於 $1 \le i \le P$，$1 \le T_i \le S$
$1 \le Q \le 10^5$
對於 $1 \le i \le Q-1$，$1 \le D_i < D_{i+1} \le 10^9$

	佔分	約束條件
1	10	$L \le 100$ $D_Q \le 100$
2	8	$P = 1$ 對於 $1 \le i \le N$，$U_i = V_i$
3	13	$P = 1$ 對於 $1 \le i \le N$，$V_i - U_i + 1 \le S$
4	19	$P = 1$
5	22	$N = 1$
6	28	無額外約束

J264 - 害羞的學生

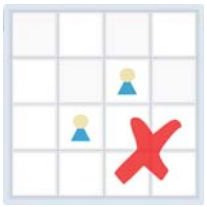
執行時間限制: 1.000 s / 記憶體限制: 256 MB

作為位元組國中學的新老師，你意識到你的班級完全由害羞的學生組成。因此，設計一個讓所有人都滿意的座位表簡直是一場噩夢...

在位元組國中學，所有教室都是由 4000 行和 4000 列組成的網格。身為教師，你知道實際上可能不需要那麼大的空間來安排學生。因此，你的教室座位表不一定要使用所有的 4000 行和列，而且有些格子可以保持為空格。

在你的座位表中，如果兩個格子 (r_1, c_1) 和 (r_2, c_2) 共享一條邊，則它們被認為是 相鄰 的。正式地，這表示為 $|r_1 - r_2| + |c_1 - c_2| = 1$.

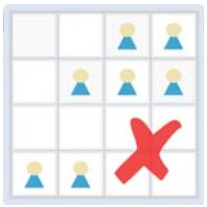
圖一：共享一條邊的兩個格子是相鄰的



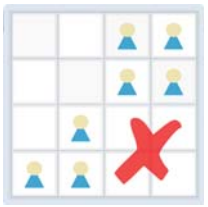
圖二: 只共享一個角的兩個格子**不是**相鄰的

你的班級總共有 N 名學生。在一個座位表中，N 名學生中的每一位都被指定在教室網格上的一個獨特格子。為了使座位表被視為 **有效**，學生必須是 連通 的。當且僅當任何兩名學生都可以通過重複移動到被學生佔據的相鄰格子來到達彼此的格子時，學生才被認為是連通 的。

圖三：連通的學生



圖四：**不** 連通的學生

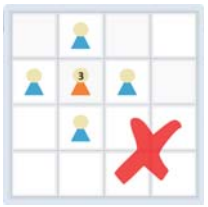


圖五：**不** 連通的學生

每位學生都有一個特定的害羞值 s，其中 $1 \le s \le 4$。一個害羞值為 s 的學生，當且僅當他們相鄰於至多 s 位其他學生時，他們才感到滿意。例如，一位害羞值為 3 的學生可以與 0、1、2 或 3 位其他學生相鄰，但不能與 4 位其他學生相鄰。

圖六：害羞值為 3 的學生最多可以與 3 位其他學生相鄰而坐

圖七：害羞值為 3 的學生最多可以與 3 位其他學生相鄰而坐



圖八：害羞值為 3 的學生 **不能** 與超過 3 位其他學生相鄰而坐

目前，你知道害羞值為 1, 2, 3 和 4 的學生人數分別為 N_1, N_2, N_3 和 N_4（其中 $N_1 + N_2 + N_3 + N_4 = N$）。你的任務是決定維度 $R \times C$ 並設計一個有效座位表，其中包含所有 N 位學生的格子，並且所有害羞的學生都感到滿意。請注意 R 和 C 都應滿足 $1 \le R, C \le 4000$。若不存在這樣的座位表，你應報告此情況。

輸入

輸入的第一行也是唯一一行包含四個整數 N_1, N_2, N_3 和 N_4。

輸出

如果不存在能滿足所有學生的有效座位表，則輸出 `Impossible`。

否則，在第一行輸出 `Possible`。在第二行輸出座位表的維度 R 和 C。然後輸出 R 行，每行包含 C 個字符，表示座位表。

在座位表中，字符 `.` 表示沒有被任何學生佔據的格子，而字符 `1`, `2`, `3`, `4` 表示佔據該座位的學生的害羞值。

如果存在多個座位表，你可以輸出其中任何一個。請注意，你的座位表的維度不必最小化，而只需要滿足 $1 \le R, C \le 4000$。

樣例

	輸入	輸出

1	`3 3 1 2`	`Possible`
		`2 8`
		`....1...`
		`12223441`

害羞值為 3 的學生與 3 位其他的學生相鄰，因此該學生感到滿意。
害羞值為 4 的兩位學生與 2 位其他的學生相鄰，並不超越 4，因此這兩位學生也感到滿意。
可以看到所有學生都感到滿意，因此這是一個有效的座位表。

2	`2 1 7 7`	`Possible`
		`4 7`
		`.444...`
		`244441.`
		`.33.3..`
		`13333..`

3	`4 1 0 0`	`Impossible`

可以證明不存在一個使所有學生同時滿意且相連的座位表。

子任務

對於所有數據:
$0 \le N_1, N_2, N_3, N_4 \le 1000$
$1 \le N_1 + N_2 + N_3 + N_4 \le 4000$

	佔分	約束條件
1	9	$N_1 = 0$
2	8	$N_1 \le 2$
3	17	$N_2 = N_3 = 0$
4	19	$N_3 = 0$
5	20	$N_2 = 0$
6	27	無額外約束