2024 年全國資訊學科能力競賽機器測試

2024.12.06

答題說明

1. 本次測驗所有題目輸出、輸入皆以標準輸入/輸出為之。
2. 每一題之執行時限限制與記憶體限制以線上裁判系統顯示的設定值為準。

考試前請確認手機已關機。

請等待監考人員宣佈測驗開始才翻頁作答。

（此頁為空白頁）

A. 共線點群 (Collinear)

問題描述

在二維平面上給定 n 個相異點，要判斷是否可以從這 n 個點，找出 k 個點同在一條直線上，而且 $k \geq \frac{n}{4}$。

這個問題我們可以這樣做：枚舉任兩個點 x 和 y，花 $O(n)$ 的時間，判斷其他 $n-2$ 個點哪些落在 x 和 y 所決定的直線上。若存在那 k 個點同在一條直線上，在枚舉的過程中會發現所花的總時間為 $O(n^3)$，可想而知，該做法雖然想法很單純，但是執行時間太慢。

我們也可以這樣做：隨機從 n 個點中選取兩個相異點 x 和 y，花 $O(n)$ 的時間，判斷其他 $n-2$ 個點哪些落在 x 和 y 所決定的直線上。若那 k 個點存在，因為 $k \geq \frac{n}{4}$，x 和 y 同時落在這 k 個點中的機率為

$$p = \frac{\binom{k}{2}}{\binom{n}{2}} \geq \frac{1}{16} - O\left(\frac{1}{n}\right).$$

我們可以重複操作上面的動作 $s = O(\log n)$ 次，每次都失敗的機會為

$$(1-p)^s = \frac{1}{n^{\Omega(1)}}.$$

根據上面的操作，我們所花的總時間為 $O(n \log n)$，而當 n, s 足夠大時，失敗機率非常地小。所以我們可以在 n 小的時候，使用第一個演算法，避免出錯，在 n 大的時候，使用第二個演算法，提高計算速度。

- 註：隨機選取可以使用 `(rand() % n) + 1` 的方式，就可以以隨機的方式回傳一個介於 1 至 n 的整數，也可以使用 `srand(time(0))`，利用時間函數重設隨機序列，避免測資針對預設的 `rand()` 函式做特殊處理。
  - 不過，`rand()` 函式提供的隨機函數並不均勻，若想要使用更均勻的函數，可以參考 `std::mt19937` 等函式庫。

輸入格式

$n\ \ k$
$x_1\ \ y_1$
$x_2\ \ y_2$
$\vdots$
$x_n\ \ y_n$

- n 為給定點的數量。
- k 為要求找出同在一條直線上的點數量。
- x_i, y_i 代表給定的第 i 個點。

輸出格式

如果存在 k 個點同在一條直線上，請輸出

```
Yes
```

否則請輸出

```
No
```

測資限制

- $1 \leq n \leq 10^5$。
- $\frac{n}{4} \leq k \leq n$。
- $-10^9 \leq x_i, y_i \leq 10^9$。
- 保證任兩個給定的點相異。
- 輸入的數皆為整數。

範例測試

Sample Input	Sample Output
5 3 2 3 2 5 4 4 5 2 6 5	Yes
5 4 2 3 2 5 4 4 5 2 6 5	No

評分說明

本題共有三組子任務，條件限制如下所示。每一組可有一或多筆測試資料，該組所有測試資料皆需答對才會獲得該組分數。

子任務	分數	額外輸入限制
1	30	$k = n$。
2	10	$n \leq 300$。
3	60	無額外限制。

（此頁為空白頁）

B. 隱藏的排列 (Permutation)

本題為互動題，限用 C++ 上傳。

問題描述

Alice 和 Bob 正在遊玩猜謎遊戲，由 Alice 負責出題目、Bob 負責猜謎。

遊戲過程如下：

- Alice 會先在心中想好一個 $1 \sim n$ 的排列。換句話說，Alice 心中有一個隱藏的序列 $p_1, p_2, \ldots, p_n$，滿足這些數字都介在 $1 \sim n$ 之間，且每個數字出現恰好一次。
- 接著，Bob 可以詢問 q 個問題，每個問題都會是以「請問 p_i 和 p_j 兩個數字誰比較大？」這種形式呈現。Alice 在收到問題後，必須如實回答。
- 在 Bob 問完所有問題後，Alice 會問 Bob k 個問題，每個問題都會是以「請問 p_i 是多少？」這種形式呈現。Bob 在收到問題後，必須給出正確的回答。

這個遊戲的目的就是要讓 Bob 詢問的問題數量 q **儘量小**來使得 Bob 能正確回答出 Alice 的所有詢問。請協助 Bob，在 q 儘量小的情況下，正確回答所有 Alice 的 k 個問題。

實作細節

你需要實作兩個函式 `bob_init()` 與 `query_from_alice()`：

```
void bob_init(int n);
```

- 對於每一筆測試資料，正式評分程式會呼叫你實作的 `bob_init()` 函式恰好 1 次。
- n 代表 Alice 心中想著的排列長度。

```
int query_from_alice(int a);
```

- 對於每一筆測試資料，正式評分程式會呼叫你實作的 `query_from_alice()` 函式恰好 k 次。
- 保證在呼叫完 `bob_init()` 後才會呼叫此函式。
- `query_from_alice()` 需要回傳一個整數 x，代表 p_a 的實際數值。

此外，在實作 `bob_init` 時可以呼叫 `compare_numbers()` 這個函式。

```
bool compare_numbers(int a, int b);
```

- a 是一個介於 $1 \sim n$ 的整數。
- b 是一個介於 $1 \sim n$ 的整數。
- $a \neq b$。
- 若 $p_a < p_b$，則該函式會回傳布林值 `true`，否則會回傳布林值 `false`。
- **範例評分程式**內的 `compare_numbers()` 實作與**實際評分程式**內的實作完全相同。

互動範例

一個可能被評為 Accepted 的互動例子顯示如下：

評分程式端	參賽者端
呼叫 bob_init(3)。	
	呼叫 compare_numbers(1, 2)。
回傳 false。	
	呼叫 compare_numbers(1, 3)。
回傳 true。	
	回傳 void()。
呼叫 query_from_alice(1)。	
	回傳 2。
呼叫 query_from_alice(2)。	
	回傳 1。

測資限制

- $1 \leq n \leq 1\,000$。
- $1 \leq k \leq 1\,000$。

範例評分程式

範例評分程式採用以下格式輸入：

```
n k
p_1 p_2 ... p_n
a_1 a_2 ... a_k
```

請注意，正式的評分程式一定不會採用以上格式輸入。請不要自行處理輸入輸出。

範例評分程式首先呼叫 bob_init(n)，接著範例評分程式會呼叫 k 次 query_from_alice(a_i)。接著，若範例評分程式偵測到從 bob_init 對 compare_numbers 的呼叫有任何不合法、或在 query_from_alice 的期間有對 compare_numbers 的呼叫，此程式將輸出

```
Wrong Answer: msg
```

後並終止程式執行，其中 msg 為下列其中之一錯誤訊息：

- Invalid position: v: 你的程式傳入 compare_numbers 的集合中有不介在 $1 \sim n$ 之間的數字 v。

- `Identical position: v`: 你的程式傳入 `compare_numbers` 的兩個數字相同。
- `Invalid call`: 你的程式嘗試在 `query_from_alice` 的期間呼叫 `compare_numbers`。

否則，範例評分程式將會以下列格式印在標準輸出中：

```
b_1 b_2 ... b_k
Accepted: Q
```

其中，

- b_i 為第 i 次呼叫 `query_from_alice()` 時你的回傳值。
- Q 為根據你的程式呼叫 `compare_numbers` 的次數得來的數值，詳細定義請見評分說明欄位。
- 請注意，**範例評分程式並不會幫助你檢查你回傳的數值是否正確**。

評分說明

對於每一筆測試資料，若你的程式在函式 `bob_init()` 中呼叫 `compare_numbers` 的次數為 x，則定義 Q 為：

$$Q = \left\lceil \frac{x}{n} \right\rceil$$

根據 Q ，你將得到分數比重 W：

$$W = \begin{cases} 1.0 & \text{若 } Q \leq 9\,; \\ \frac{3}{\sqrt{Q}} & \text{若 } 9 < Q \leq 500\,; \\ 0 & \text{若 } Q > 500。 \end{cases}$$

本題共有兩組子任務，條件限制如下所示。每一組可有一或多筆測試資料，你在該子任務的得分為所有測試資料中分數比重 W 的最小值，乘以該子任務的總分。

子任務	分數	額外輸入限制
1	10	$n = 2$。
2	90	無額外限制。

C. 數獨 (Sudoku)

本題為 Output Only。請注意，你在第 i 筆測試資料上傳的輸出檔名必須為 `output_i_1.txt`。

問題描述

費里敦（Fereydun）是一位傳說中的波斯英雄，根據預言，他將戰勝扎哈克（Zahhak）。費里敦相信，除了強壯的體魄外，還需要擁有強大的頭腦。他最近從一位日本商人那裡學到了一個名為數獨（Sudoku）的益智遊戲。

數獨是在一個 $n^2 \times n^2$ 的棋盤上進行的遊戲。整個棋盤被分成 n^2 個子區域，每個子區域的大小為 $n \times n$。每個格子可以是空的，也可以包含一個從 1 到 n^2 之間的整數。一個數獨棋盤是合法的，需滿足以下條件：

1. 每一行中的所有數字都是不同的。
2. 每一列中的所有數字都是不同的。
3. 每個子區域中的所有數字都是不同的。

以下圖示顯示了兩個沒有空格子的合法數獨棋盤：

1	2	3	4
3	4	1	2
2	1	4	3
4	3	2	1

1	2	3	4	5	6	7	8	9
4	5	6	7	8	9	1	2	3
7	8	9	1	2	3	4	5	6
2	3	1	5	6	4	8	9	7
5	6	4	8	9	7	2	3	1
8	9	7	2	3	1	5	6	4
3	1	2	6	4	5	9	7	8
6	4	5	9	7	8	3	1	2
9	7	8	3	1	2	6	4	5

數獨棋盤範例圖

費里敦有一個帶有一些空格的合法數獨棋盤，並請求你的幫助。你的任務是**儘可能多**地填充這些空格，同時保持棋盤有效。

保證至少存在一種方法可以填滿所有空格且使棋盤保持合法。

輸入格式

$$
\begin{array}{l}
n \\
a_{1,1}\ \ a_{1,2}\ \ \cdots\ \ a_{1,n} \\
a_{2,1}\ \ a_{2,2}\ \ \cdots\ \ a_{2,n} \\
\vdots \\
a_{n,1}\ \ a_{n,2}\ \ \cdots\ \ a_{n,n}
\end{array}
$$

- n 為棋盤的大小。
- $a_{i,j}$ 代表棋盤第 i 列第 j 欄格子內的數字。

輸出格式

$$
\begin{array}{l}
b_{1,1}\ \ b_{1,2}\ \ \cdots\ \ b_{1,n} \\
b_{2,1}\ \ b_{2,2}\ \ \cdots\ \ b_{2,n} \\
\vdots \\
b_{n,1}\ \ b_{n,2}\ \ \cdots\ \ b_{n,n}
\end{array}
$$

- $b_{i,j}$ 代表你給出的棋盤中，第 i 列第 j 欄格子內的數字。

測資限制

- $2 \leq n \leq 20$。
- $0 \leq a_{i,j} \leq n^2$。
- 測試資料保證存在一種可以填滿所有空格且使棋盤保持合法的方式。

評分說明

本題共有 10 組測試資料，輸入檔案的說明如表所示。對於每一組測試資料，若你上傳的輸出檔案滿足輸出格式且為合法的數獨棋盤，那麼你會得到以下分數

$$\frac{10 \times (p - q)}{p}$$

其中 p 是最初棋盤的空格數量、q 是你給出的棋盤內的空格數量。

若你上傳的輸出檔案不滿足輸出格式、有原本非空的格子被填入了不同的數字、或是你輸出的棋盤不是合法的數獨棋盤，那麼你將得到 0 分。

測試資料	分數	輸入檔名	輸出檔名	說明
1	10	`input_1_1.txt`	`output_1_1.txt`	$n = 2$。
2	10	`input_2_1.txt`	`output_2_1.txt`	$n = 3$。
3	10	`input_3_1.txt`	`output_3_1.txt`	$n = 3$。
4	10	`input_4_1.txt`	`output_4_1.txt`	$n = 10$。
5	10	`input_5_1.txt`	`output_5_1.txt`	$n = 20$。
6	10	`input_6_1.txt`	`output_6_1.txt`	$n = 4$。
7	10	`input_7_1.txt`	`output_7_1.txt`	$n = 8$。
8	10	`input_8_1.txt`	`output_8_1.txt`	$n = 12$。
9	10	`input_9_1.txt`	`output_9_1.txt`	$n = 16$。
10	10	`input_10_1.txt`	`output_10_1.txt`	$n = 20$。

範例

作為範例，假設測試資料的長相為

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

則下面是一個可能的合法輸出

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

在這個範例中，p 和 q 的值分別為 12 和 2。因此，這個輸出可以獲得 $\frac{10\times(12-2)}{12} \sim 8.33$ 分

視覺化工具（Visualizer）

為了方便選手觀看自己的輸出結果以及觀察測試資料，在此任務的附件（attachment）中，有一腳本程式（script）供選手視覺化（visualize）輸入檔與輸出檔。

請利用下列指令視覺化輸入檔。因為技術上的限制，附件中提供的視覺化工具僅能展示 $n \leq 4$ 的數獨棋盤。

```
python3 visualizer.py [input file]
```

你可利用下列指令，將你對於某個輸入計算出的解做視覺化。

```
python3 visualizer.py [input file] --solution [output file]
```

範例：

```
python3 visualizer.py input_1_1.txt --solution output_1_1.txt
```

請注意，若你傳入的資料的格式並不合法，將會產生一些不可預期的行為。