# จดโพยข้อสอบ

ในปี 3202 คุณได้มีโอกาสเป็นผู้แทนประเทศเข้าแข่งขัน the 81st International Olympiad in Trickmath (81IOT) แต่เนื่องจากคุณไม่ชอบทริคแมท คุณจึงตัดสินใจที่จะ*โกง*ในการแข่งขันครั้งนี้

ข้อสอบในการแข่งขันเป็นข้อสอบเติมคำตอบ $N$ ข้อ แน่นอนว่าคุณได้ยัดเงินทีมออกโจทย์เพื่อเอาคำตอบมาหมดแล้ว และรู้มาว่าคำตอบของข้อสอบทุกข้อมีค่าตั้งแต่ $0$ ถึง $K$ ซึ่งอาจมีค่าซ้ำกันได้ อย่างไรก็ตาม ถ้าคุณจดโพยเข้าไปตรง ๆ กรรมการคุมสอบก็จะจับได้ว่ามีคนโกง คุณจึงเหลือบไปเห็นสำรับไพ่ตัวเลขที่ระบุค่าเรียงจาก $0$ ถึง $1\,000\,000$ และมีไอเดียที่จะนำไพ่เหล่านี้เข้าห้องสอบไปแทนโพยที่มีอยู่

ดังนั้นแล้ว คุณจึงวางแผนสร้างโพยข้อสอบขึ้นมาจากการหยิบไพ่บางใบมาเรียงกันเป็นลำดับใหม่ กล่าวคือโพยชิ้นใหม่ของคุณจะเป็นลำดับของ**ตัวเลขที่ไม่ซ้ำกันเลย** ซึ่งคุณต้องสามารถใช้มันในการตอบคำถามให้ถูกต้องได้ทุกข้อ

นอกจากนี้แล้ว ไพ่ที่ระบุตัวเลขค่ามาก ๆ จะกินพื้นที่ในกระเป๋าของคุณ การสร้างลำดับไพ่ขึ้นมาใหม่จึงควรทำให้**ค่ามากสุดของลำดับ**มีค่าต่ำที่สุดที่เป็นไปได้ด้วยเช่นกัน

## รายละเอียดการเขียนโปรแกรม

คุณจะต้องเขียนฟังก์ชันสองฟังก์ชันต่อไปนี้

```
vector<int> write_cheatsheet(int N, vector<int> A)
```

- ฟังก์ชันนี้จะถูกเรียกใช้โดยเกรดเดอร์เพียงครั้งเดียว
- $N$: จำนวนข้อทั้งหมดของข้อสอบ
- $A$: คำตอบของข้อสอบแต่ละข้อ โดยที่ $0 \le A[i] \le K$ สำหรับ $0 \le i < N$
- ฟังก์ชันจะต้องคืนอาเรย์ $R$ ซึ่งเป็นอาเรย์ที่มีสมาชิกไม่ซ้ำกัน โดยขนาดของอาเรย์จะต้องมีค่าไม่เกิน $1\,000\,000$ และ $0 \le R[j] \le 1\,000\,000$ สำหรับ $0 \le j < R.size()$

```
vector<int> recover_answer(int N, vector<int> R)
```

- ฟังก์ชันนี้จะถูกเรียกใช้โดยเกรดเดอร์เพียงครั้งเดียว
- $N$: จำนวนข้อทั้งหมดของข้อสอบ
- $R$: อาเรย์ของโพยข้อสอบที่ผู้เข้าแข่งขันส่งมาจากฟังก์ชันแรก
- ฟังก์ชันจะต้องคืนอาเรย์ขนาด $N$ โดยในตำแหน่งที่ $i$ จะเป็นค่าคำตอบของข้อสอบข้อที่ $i$ เรียงตามลำดับที่ได้รับมาในตอนแรก

ในแต่ละชุดทดสอบจะมีการเรียกใช้แต่ละฟังก์ชันด้วยค่า $N$ และอาเรย์ $A$ เพียงครั้งเดียวผ่านการรันโปรแกรมทั้งหมดสองครั้ง ดังนี้

ในการรันโปรแกรมครั้งแรก

- ฟังก์ชัน `write_cheatsheet` จะถูกเรียกใช้เพียงครั้งเดียว และส่งอาเรย์ที่ได้รับไปยังการรันโปรแกรมครั้งที่สอง

ในการรันโปรแกรมครั้งที่สอง

- ฟังก์ชัน `recover_answer` จะถูกเรียกใช้เพียงครั้งเดียว

ซึ่งในการรันโปรแกรมครั้งที่สอง ตัวแปร static หรือ global ที่ได้ประกาศไว้ในการรันโปรแกรมครั้งแรกจะไม่สามารถถูกนำมาใช้งานได้

# เงื่อนไข

- $1 \leq N \leq 500$
- $0 \leq K \leq 350\,000$

# ปัญหาย่อย

1. (5 คะแนน) $N = 80, K = 12\,500$
2. (95 คะแนน) $N = 500, K = 350\,000$

ในปัญหาย่อยที่ 2 จะมีการให้คะแนนบางส่วน กำหนดค่า $M$ แทนค่าสูงสุดของสมาชิกทั้งหมดในอาเรย์ $R$ กล่าวคือ $M = max(R[j])$ สำหรับ $0 \leq j < R.size()$ สังเกตว่า**ขนาดของอาเรย์ $R$ จะไม่มีผลต่อการให้คะแนน** ทั้งนี้คะแนนที่ได้จะถูกประมาณเป็นทศนิยมสองตำแหน่ง

| เงื่อนไข | คะแนนที่ได้ |
|---|---|
| $10^6 < M$ | $0$ |
| $350\,000 < M \leq 10^6$ | $20 - \frac{M}{50\,000}$ |
| $4500 < M \leq 350\,000$ | $30 - \frac{M - 4500}{34550}$ |
| $3500 < M \leq 4500$ | $40 - \frac{M - 3500}{100}$ |
| $1600 < M \leq 3500$ | $60 - \frac{M - 1600}{95}$ |
| $1550 < M \leq 1600$ | $80 - \left(20 - \frac{(1600 - M)^2}{125}\right)$ |
| $1067 < M \leq 1550$ | $95 - \frac{5(M - 1067)}{161}$ |
| $M \leq 1067$ | $95$ |

## ตัวอย่าง

พิจารณาการเรียกใช้ฟังก์ชันข้างต้น

```
write_cheatsheet(5, [1, 2, 1, 4, 0])
```

ในการเรียกใช้ครั้งนี้จะมีข้อสอบทั้งหมด $5$ ข้อ โดยแต่ละข้อมีคำตอบเท่ากับ $1, 2, 1, 4, 0$ ตามลำดับ ทั้งนี้การคืนอาเรย์ $R$ จากฟังก์ชันนี้ไม่จำเป็นที่จะต้องมีขนาดเท่ากับอาเรย์ $A$ เสมอไป โดยสมมุติว่าฟังก์ชันดังกล่าวคืนค่าเป็น $[10, 7, 1, 5, 8, 2, 4]$

เมื่อเรียกใช้ฟังก์ชันข้างต้นเสร็จแล้ว ในการรันโปรแกรมครั้งที่สองจะมีการเรียกใช้ฟังก์ชันดังนี้

```
recover_answer(5, [10, 7, 1, 5, 8, 2, 4])
```

ฟังก์ชันนี้จะต้องคืนอาเรย์ $[1, 2, 1, 4, 0]$ ซึ่งเป็นคำตอบของข้อสอบที่ได้รับมาในครั้งแรก ในที่นี้จะพบว่า $M$ (ค่าสูงสุดของสมาชิกในอาเรย์ $R$) จะมีค่าเท่ากับ $10$

## เกรดเดอร์ตัวอย่าง

เกรดเดอร์ตัวอย่างจะอ่านข้อมูลนำเข้าดังนี้

- บรรทัดที่ $1$: $N$
- บรรทัดที่ $2$: $A[0]$ $A[1]$ $A[2]$ $\ldots$ $A[N-1]$

ในกรณีที่คำตอบถูกต้อง เกรดเดอร์ตัวอย่างจะพิมพ์จำนวนเต็ม $M$

ในกรณีที่คำตอบไม่ถูกต้อง เกรดเดอร์ตัวอย่างจะพิมพ์ `Wrong answer: <MSG>` โดยที่ `<MSG>` จะเป็นหนึ่งในข้อความข้างต้น

- `size limit exceeded`: ขนาดของอาเรย์ $R$ มีค่าเกิน $1\,000\,000$
- `duplicate elements`: สมาชิกในอาเรย์ $R$ มีค่าที่ซ้ำกัน
- `value out of bound`: สมาชิกในอาเรย์ $R$ มีค่าน้อยกว่า $0$ หรือมากกว่า $1\,000\,000$
- `incorrect answer size`: อาเรย์คำตอบมีขนาดไม่ตรงกับอาเรย์ $A$
- `incorrect value`: อาเรย์ที่ได้จากการเรียกใช้ฟังก์ชัน `recover_answer` ไม่ตรงกับอาเรย์ $A$

## ขีดจำกัด

- Time limit: 1 second
- Memory limit: 512 MB