ขายของออนไลน์

ในการประมูลซื้อของออนไลน์ของรูปวาดดิจิทัลทรงคุณค่าครั้งหนึ่ง ผู้ขายตระหนักว่าเนื่องจากของเป็นดิจิทัล การขาย copy หนึ่งในราคาเท่าใดก็ได้กำไรทั้งนั้น แต่การที่ผู้ซื้อเข้ามาดูราคาแล้วพบว่าราคาที่เสนอแพงกว่าราคาที่ผู้ซื้ออยากได้ (ราคาที่ผู้ซื้อให้มูลค่าเอาไว้) ผู้ซื้อก็จะไม่ซื้อสินค้านั้น

พิจารณาเหตุการณ์ต่อไปนี้ สมมติว่าผู้ซื้อคนหนึ่งให้มูลค่ารูปวาดไว้ 15 บาท

- ถ้าผู้ขายเสนอราคา 1 บาท ผู้ซื้อตกลงซื้อ จ่ายเงิน 1 บาท
- ถ้าผู้ขายเสนอราคา 10 บาท ผู้ซื้อตกลงซื้อ จ่ายเงิน 10 บาท
- ถ้าผู้ขายเสนอราคา 15 บาท ผู้ซื้อตกลงซื้อ จ่ายเงิน 15 บาท
- ถ้าผู้ขายเสนอราคา 16 บาท ผู้ซื้อไม่ตกลงซื้อ จ่ายเงิน 0 บาท
- ถ้าผู้ขายเสนอราคา 80 บาท ผู้ซื้อไม่ตกลงซื้อ จ่ายเงิน 0 บาท

เพื่อความชัดเจน สมมติว่าผู้ซื้อให้มูลค่า V และเราเสนอราคา P เราจะได้เงิน P บาท ถ้า $P \leq V$ แต่เราจะได้เงิน 0 บาท ถ้า $P > V$

สังเกตว่าถ้าเราทราบมูลค่าในใจ เราจะสามารถหารายได้รวมได้เยอะที่สุด แต่ถ้าเราไม่ทราบ การเสนอราคาที่ต่ำอาจจะทำให้ได้เงินบ้าง แต่อาจจะได้น้อยมากกว่าที่สามารถทำได้

คุณมีผู้ซื้อ $100\,000$ คน แต่ละคนมีมูลค่าสินค้าที่ไม่จำเป็นต้องเท่ากัน โดยมีค่าเป็นจำนวนเต็มระหว่าง $1\,000 - 1\,000\,000$ มิลลิบาทซึ่งถูกระบุไว้ล่วงหน้า ผู้ซื้อจะเข้ามาในระบบของคุณในลำดับแบบสุ่ม (แต่ในการตรวจโปรแกรมของคุณลำดับในชุดข้อมูลทดสอบนั้นจะคงที่แต่เป็นลำดับที่ได้มาจากการสุ่มแล้ว) เมื่อเจอผู้ซื้อแต่ละคน คุณจะต้องเสนอราคาให้กับผู้ซื้อและจะทราบผลว่าผู้ซื้อตกลงซื้อหรือไม่ คุณไม่จำเป็นต้องเสนอราคาเท่ากันทุกคน

ให้เขียนโปรแกรมเพื่อเสนอราคาขายให้ได้รายได้รวมสูงที่สุด (กรุณาอ่านรายละเอียดเกี่ยวกับการทดสอบและการทดลองปรับพารามิเตอร์ของโปรแกรมกับข้อมูลทดสอบในส่วนการให้คะแนน)

รายละเอียดการเขียนโปรแกรม

คุณจะต้องเขียนฟังก์ชันต่อไปนี้

```
void sell_all(int N)
```

- ฟังก์ชันนี้จะถูกเรียกหนึ่งครั้ง และจะเป็นฟังก์ชันหลักที่คุณจะต้องเขียน เพื่อขายสินค้า
- คุณจะได้รับ N เป็นจำนวนลูกค้า ในการตรวจ $N = 100\,000$ เสมอ แต่ที่ส่งให้สำหรับฟังก์ชันนี้ มีไว้ให้คุณสามารถทดสอบโปรแกรมได้สะดวก โดยสามารถเปลี่ยนค่า N ได้ที่เกรดเดอร์ตัวอย่าง
- ฟังก์ชันนี้จะเรียกฟังก์ชัน `sell_price` เป็นจำนวน N ครั้ง เพื่อเสนอราคาสินค้า
- เมื่อเรียกครบแล้วให้จบการทำงานของฟังก์ชัน

ฟังก์ชัน `sell_all` จะต้องเรียกฟังก์ชันต่อไปนี้จำนวน N ครั้ง (ในเกรดเดอร์จริง $N = 100\,000$ เสมอ)

```
bool sell_price(int P)
```

- การเรียกฟังก์ชันนี้คือการเสนอราคาขาย P มิลลิบาทให้กับผู้ซื้อ
- ค่าที่คืนจากฟังก์ชันจะระบุว่าผู้ซื้อซื้อสินค้าหรือไม่ ถ้าผู้ใช้ซื้อจะคืนค่า `true` ถ้าไม่ซื้อจะคืนค่า `false`

ขอบเขต

- จำนวนลูกค้ามี $100\,000$ คนพอดีเสมอ
- มูลค่าที่ลูกค้าแต่ละคนให้กับสินค้ามีค่าระหว่าง $1\,000$ ถึง $1\,000\,000$ มิลลิบาท
- $0 \le P \le 1\,000\,000$ (ขอบเขตราคาขาย)
- $1\,000 \le V[i] \le 1\,000\,000$ (มูลค่าที่ลูกค้าคนที่ i ให้กับสินค้า)

ปัญหาย่อย

แต่ละปัญหาย่อยของโจทย์ข้อนี้คือรูปแบบของการกระจายของมูลค่าสินค้าที่ลูกค้าแต่ละคนให้ที่แตกต่างกัน โปรแกรมที่ส่งของคุณจะต้องทำงานได้กับทุก ๆ การกระจายในแต่ละปัญหาย่อยพร้อม ๆ กัน (นั่นคือคะแนนที่คุณจะได้คือคะแนนของ submission ที่ทำคะแนนรวมทุกปัญหาย่อยมากที่สุด -- ดูส่วนการตรวจและให้คะแนนเพิ่มเติม)

การตรวจให้คะแนน และการทดลองกับข้อมูลทดสอบ

ในเกรดเดอร์จะมีโจทย์ให้ส่ง 3 ข้อ สำหรับทดลองส่ง 2 ข้อ สำหรับส่งจริง 1 ข้อ ข้อมูลทดสอบของโจทย์ทั้งสามข้อจะสร้างจากการกระจายแบบเดียวกัน ดังนั้นโปรแกรมควรจะทำคะแนนได้ใกล้เคียงกันในชุดข้อมูลเหล่านี้ โจทย์ทดสอบสองข้อ ห้ามส่งเกินข้อละ 50 ครั้ง **สำหรับโจทย์ส่งจริงจะส่งได้ไม่เกิน 20 ครั้ง เพื่อป้องกันการ reverse engineer ข้อมูลทดสอบ** ถ้าลืมส่งที่โจทย์ส่งจริง สามารถแจ้งให้เอา submission จากโจทย์ทดลองส่งมาส่งได้หลังแข่ง

เพื่อป้องกันโอกาสโชคร้ายสุด ๆ จะมีชุดทดสอบพิเศษซ่อนไว้ (สร้างจากการกระจายแบบเดียวกันเช่นกัน) สำหรับให้คะแนนอีกหนึ่งข้อ ที่จะไม่ให้ส่ง แต่จะนำมาตรวจด้วยเมื่อให้คะแนน ถ้าได้คะแนนมากกว่าข้อส่งจริง จะให้คะแนนจากข้อที่มีชุดทดสอบพิเศษนี้

การคิดคะแนน รายได้ที่คุณทำได้จะถูกเทียบกับรายได้สูงสุดที่โปรแกรมของกรรมการทำได้ (ไม่ใช่เทียบกับของผู้สอบด้วยกัน เพราะว่าอาจจะมีบางโปรแกรมทำมูลค่าได้สูงมาก ๆ กับบางกรณีทดสอบเท่านั้น การเทียบกับค่าสูงสุดแยกตามกรณีทดสอบดังกล่าวจะไม่ยุติธรรม) รายได้สูงสุดที่โปรแกรมของกรรมการทำได้มีการประกาศให้ผู้เข้าสอบทราบ นอกจากนี้ในตารางจะมีคอลัมน์รายได้ของโปรแกรมที่โกง (judge-cheat) โดยการทราบการกระจายข้อมูลก่อน **ค่านี้มีไว้เพื่อการสังเกตเฉย ๆ ไม่ได้นำมาเทียบกับโปรแกรมของผู้เข้าแข่งขัน**

ถ้าคุณทำรายได้ได้มากกว่าหรือเท่ากับ 90% ของรายได้ที่โปรแกรมของกรรมการทำได้ (คิดจากรายได้ที่โปรแกรมได้น้อยที่สุดของทั้งสามชุดทดสอบ) คุณจะได้คะแนนเต็ม ไม่เช่นนั้น คุณจะได้คะแนนลดหลั่นไปตาม scale ของ 90% ของรายได้ที่โปรแกรมกรรมการทำได้ โดยใช้สมการ 1/อัตราส่วนยกกำลัง 2 (ถ้า

คลาดเคลื่อนไปสองเท่า จะได้คะแนน 25%)

สำหรับแต่ละกรณีทดสอบ ระบบทดสอบจะแสดงผลลัพธ์เป็น `(testcase:tvalue/score)` โดย `testcase` เป็นหมายเลขกรณีทดสอบ และ `tvalue` คือรายได้รวมที่ขายได้ หน่วยเป็น "พันบาท" (คือมิลลิบาทหารด้วย $1\,000\,000$) และ `score` คือคะแนนที่คุณทำได้

การทดลองกับข้อมูลทดสอบ โปรแกรมของคุณอาจจะมีพารามิเตอร์บางอย่างที่สามารถปรับได้เพื่อทำให้ได้มูลค่าในการขายมากที่สุด เราเปิดให้ทดลองกับข้อมูลทดสอบของโจทย์ทดลองส่งเพื่อปรับค่าพารามิเตอร์เหล่านี้ได้ แต่ค่าพารามิเตอร์ควรจะต้องไม่ใช่ค่าเพื่อที่จะมุ่งทำงาน **อย่างเฉพาะเจาะจง** กับข้อมูลชุดทดสอบใดชุดทดสอบหนึ่ง ถ้าทดลองแบบนี้เราจะไม่นับว่าเป็นการ reverse engineer ข้อมูลทดสอบ เราจะถือว่าการใช้พารามิเตอร์และการทดลองเป็นการ reverse engineer ถ้ามีการปรับข้อมูลชุดทดสอบเพียงเล็กน้อย แล้วโปรแกรมของคุณให้ผลลัพธ์ที่แตกต่างอย่างมากจนเกินไป (เช่นปรับมูลค่าเปลี่ยนไปเพียง 10 มิลลิบาท แล้วมูลค่าที่ได้รวมเปลี่ยนไปมากมาย)

ตัวอย่าง

จะแสดงตัวอย่างที่มีลูกค้า $N = 4$ เท่านั้น การทำงานจริงจะเรียกใช้ฟังก์ชันของคุณที่ $N = 100\,000$ เสมอ

```
sell_all(4)
```

ฟังก์ชันจะต้องเรียกฟังก์ชัน `sell_price` ทั้งสิ้น $N = 4$ ครั้ง สมมติว่าลูกค้าที่มาตามลำดับ (หลังจากสุ่มแล้ว) มีมูลค่าภายในใจเป็น $5\,000$, $7\,000$, $1\,000$ และ $2\,500$ มิลลิบาทตามลำดับ

ผลลัพธ์ที่ได้จากการเรียนฟังก์ชัน `sell_price` ด้วยค่าต่าง ๆ ตามลำดับจะเป็นดังตารางนี้

ครั้งที่	การเรียกฟังก์ชัน	ค่าที่คืนกลับมา	คำอธิบาย	เงินที่ได้	เงินที่ได้รวม
1	`sell_price(500)`	`true`	มูลค่าของลูกค้ามากกว่า 500 เลยตกลงซื้อด้วยราคา 500	500	500
2	`sell_price(10000)`	`false`	มูลค่าของลูกค้าน้อยกว่า 10 000 เลยไม่ซื้อ	0	500
3	`sell_price(1001)`	`false`	มูลค่าของลูกค้าน้อยกว่า 1 001 เลยไม่ซื้อ	0	500
4	`sell_price(2500)`	`true`	มูลค่าของลูกค้าเท่ากับ 2 500 เลยตกลงซื้อด้วยราคา 2 500	2 500	3 000

เมื่อจบการทำงาน คุณจะขายสินค้าได้มูลค่ารวม $3\,000$ มิลลิบาท

เกรดเดอร์ตัวอย่าง

เกรดเดอร์ตัวอย่างอ่านข้อมูลดังนี้

- บรรทัดที่ 1: N
- บรรทัดที่ $2, \ldots, N+1$: $V[i]$ (มูลค่าสินค้าในใจของลูกค้าคนที่ i)

เกรดเดอร์จะพิมพ์รายได้รวมที่โปรแกรมทำได้

ข้อจำกัด

- Time limit: 0.25 second
- Memory limit: 512 MB