แผงตลาด

(1 วินาที, 512 MB)

ในงานฉลองวาระครบรอบ 36 ปี แห่งการสถาปนามหาวิทยาลัยเทคโนโลยีสุรนารี (มทส.) ทางมหาวิทยาลัยได้รับความร่วมมือจาก**ตลาดเซฟวัน** ในการจัดสรรพื้นที่ให้นักเรียนและนักศึกษาเข้ามาเช่าแผงขายของเพื่อจำหน่ายเหรียญที่ระลึก โดยผู้จัดการตลาดได้แบ่งพื้นที่ให้เช่าเป็นแถวยาวหนึ่งแถว ประกอบด้วย แผงขายของทั้งหมด N แผง เรียงต่อกันเป็นเส้นตรงและมีขนาดเท่ากันทุกแผง โดยกำหนดหมายเลขแผงตั้งแต่ 1 ถึง N จากซ้ายไปขวา

เมื่อถึงเวลาตลาดเปิด จะมีลูกค้าเดินทางมาซื้อเหรียญที่ระลึกงานครบรอบ 36 ปี การสถาปนา มทส. พร้อมกันเป็นจำนวน N คน โดยกำหนดให้ลูกค้าคนที่ t (เมื่อ $1 \leq t \leq N$) เดินทางมาหยุดที่หน้าแผงหมายเลข t และจะตัดสินใจซื้อเหรียญจากร้านค้าตามเงื่อนไขดังต่อไปนี้

- ลูกค้าจะซื้อเหรียญจากร้านที่ตั้งอยู่ในแผงขายของ**ที่มีระยะทางใกล้ที่สุด**กับตำแหน่งแผงที่ตนเดินเข้ามาหยุด
- หากมีร้านค้าสองร้านตั้งอยู่ด้วยระยะทางที่ห่างจากลูกค้าเท่ากันพอดี ลูกค้าจะตัดสินใจไปเลือกร้านที่มี**หมายเลขแผงสูงกว่า**

เมื่อคุณเดินทางมาถึงงาน แผงจะถูกจองไปแล้วจำนวน K แผง โดยคุณทราบว่าแผงไหนบ้างที่ถูกจองไปแล้ว ซึ่งคุณได้รับสิทธิ์ให้เช่าแผงที่ยังว่างอยู่เป็นจำนวนเท่าใดก็ได้

ค่าหวังกำไร P ของคุณคำนวณจาก

$$P = \left\lfloor \frac{C \times W}{N} \right\rfloor - T$$

โดยที่

- N คือ จำนวนแผงขายของทั้งหมดในตลาด
- W คือ กำไรต่อหนึ่งเหรียญที่ขายได้
- T คือ ค่าเช่าแผงทั้งหมดที่คุณเช่า โดยมีค่าเท่ากับจำนวนแผงทั้งหมดที่คุณเช่า
- C คือ จำนวนลูกค้าที่มาซื้อเหรียญจากร้านคุณ
- $\lfloor x \rfloor$ คือ จำนวนเต็มที่มากที่สุดที่มีค่า $\leq x$ (**Floor Function**)

กำหนดสถานการณ์ Q แบบ แต่ละแบบ**อาจจะมีค่า** W **ที่แตกต่างกัน** แต่มีค่า N, K และตำแหน่งที่ถูกจองไปแล้ว**คงที่ในทุกสถานการณ์**

สำหรับแต่ละสถานการณ์ จงหา**ค่าหวังกำไรสูงสุดที่เป็นไปได้**

รายละเอียดการเขียนโปรแกรม

คุณจะต้องเขียน 2 ฟังก์ชัน ในไฟล์ `market.cpp` ดังนี้

```
void init(int N, vector<int> A)
```

โดย

- N: จำนวนแผงขายของทั้งหมดในตลาด
- A: vector ของหมายเลขแผงของพ่อค้าคนอื่นที่ขายเหรียญ (ค่าไม่ซ้ำกัน อาจไม่เรียงลำดับ และอาจจะไม่มีคนจองเลยก็ได้)
- **หมายเหตุ** ฟังก์ชันนี้จะถูกเรียก **1 ครั้งเท่านั้น**ต่อหนึ่งชุดทดสอบ และจะถูกเรียกก่อนการเรียก `expected_profit` (ดูเพิ่มเติมใน `grader.cpp`)

```
long long expected_profit(int W)
```

โดย

- W: กำไรต่อหนึ่งเหรียญที่ขายได้
- ฟังก์ชันนี้จะต้องคืนค่าหวังกำไรสูงสุดที่เป็นไปได้
- **หมายเหตุ** ฟังก์ชันนี้จะถูกเรียก**ทั้งหมด Q ครั้ง**

ตัวอย่างที่ 1

พิจารณาการเรียกฟังก์ชันตามนี้ ($Q = 3$)

```
init(6,[2]);
```

มีแผงขายของทั้งหมด 6 แผง โดยมีแผงที่ถูกจองไปแล้วหนึ่งแผง ได้แก่ แผงหมายเลข 2 พิจารณากรณีต่อไปนี้

สถานการณ์ที่ 1

เมื่อกำไรต่อเหรียญ $W = 10$ จะมีการเรียกดังนี้

```
expected_profit(10)
```

สมมติว่าคุณเลือกเช่าแผงหมายเลข $1, 4$ ซึ่งทำให้มีจำนวนแผงที่คุณเช่าเป็น $T = 2$

แผงที่ลูกค้าเดินเข้ามา	ระยะจากแผง 1 (คุณ)	ระยะจากแผง 2 (พ่อค้าคนอื่น)	ระยะจากแผง 4 (คุณ)	ร้านที่ลูกค้าเข้าซื้อ
1	**0**	1	3	**คุณ** (แผง 1)
2	1	**0**	2	พ่อค้าคนอื่น (แผง 2)
3	2	1	**1**	**คุณ** (แผง 4)
4	3	2	**0**	**คุณ** (แผง 4)
5	4	3	**1**	**คุณ** (แผง 4)
6	5	4	**2**	**คุณ** (แผง 4)

จะได้ว่าจำนวนลูกค้าที่เดินเข้ามาแล้วแวะร้านคุณ $C = 5$

ทำให้คิดค่าหวังกำไรได้เป็น

$$P = \left\lfloor \frac{5 \times 10}{6} \right\rfloor - 2 = 8 - 2 = \mathbf{6}$$

ซึ่งในกรณีนี้เป็นค่าคาดหวังกำไรสูงที่สุดที่เป็นไปได้แล้ว ดังนั้นฟังก์ชันจะต้องคืนค่าเป็น 6

สถานการณ์ที่ 2

เมื่อกำไรต่อเหรียญ $W = 5$ จะมีการเรียกดังนี้

```
expected_profit(5)
```

ในกรณีนี้ วิธีเลือกเช่าแผงที่ดีที่สุด คือ คุณเลือกเช่าแผงหมายเลข 3 ทำให้มีจำนวนแผงที่คุณเช่า $T = 1$

แผงที่ลูกค้าเดินเข้ามา	ระยะจากแผง 2 (พ่อค้าคนอื่น)	ระยะจากแผง 3 (คุณ)	ร้านที่ลูกค้าเข้าซื้อ
1	**1**	2	พ่อค้าคนอื่น (แผง 2)
2	**0**	1	พ่อค้าคนอื่น (แผง 2)
3	1	**0**	**คุณ** (แผง 3)
4	2	**1**	**คุณ** (แผง 3)
5	3	**2**	**คุณ** (แผง 3)
6	4	**3**	**คุณ** (แผง 3)

จะได้ว่าจำนวนลูกค้าที่เดินเข้ามาแล้วแวะร้านคุณเป็น $C = 4$

ทำให้คิดค่าหวังกำไรได้เป็น

$$P = \left\lfloor \frac{4 \times 5}{6} \right\rfloor - 1 = 3 - 1 = \mathbf{2}$$

ดังนั้นฟังก์ชันจะต้องคืนค่าเป็น 2

สถานการณ์ที่ 3

เมื่อกำไรต่อเหรียญ $W = 2$ จะมีการเรียกดังนี้

```
expected_profit(2)
```

ในกรณีนี้ วิธีเลือกเช่าแผงที่ดีที่สุด คือ คุณเลือกที่จะไม่เช่าแผงไหนเลย ทำให้ $T = 0$ และ $C = 0$ และคิดค่าหวังกำไรได้เป็น

$$P = \left\lfloor \frac{0 \times 2}{6} \right\rfloor - 0 = 0 - 0 = \mathbf{0}$$

ดังนั้นฟังก์ชันจะต้องคืนค่าเป็น 0

ตัวอย่างที่ 2

พิจารณาการเรียกฟังก์ชันตามนี้ ($Q = 1$)

```
init(6,[1,5]);
```

มีแผงขายของทั้งหมด 6 แผง โดยมีแผงที่ถูกจองไปแล้วสองแผง ได้แก่ แผงหมายเลข 1 และแผงหมายเลข 5 พิจารณากรณีต่อไปนี้

สถานการณ์ที่ 1

เมื่อกำไรต่อเหรียญ $W = 6$ จะมีการเรียกดังนี้

```
expected_profit(6)
```

ในกรณีนี้ วิธีเลือกเช่าแผงที่ดีที่สุด คือ คุณเลือกเช่าแผงหมายเลข 3 ทำให้มีจำนวนแผงที่คุณเช่า $T = 1$

แผงที่ลูกค้าเดินเข้ามา	ระยะจากแผง 1 (พ่อค้าคนอื่น)	ระยะจากแผง 3 (คุณ)	ระยะจากแผง 5 (พ่อค้าคนอื่น)	ร้านที่ลูกค้าเข้าซื้อ
1	**0**	2	4	พ่อค้าคนอื่น (แผง 1)
2	1	**1**	3	**คุณ** (แผง 3)
3	2	**0**	2	**คุณ** (แผง 3)
4	3	1	**1**	พ่อค้าคนอื่น (แผง 5)
5	4	2	**0**	พ่อค้าคนอื่น (แผง 5)
6	5	3	**1**	พ่อค้าคนอื่น (แผง 5)

จะได้ว่าจำนวนลูกค้าที่เดินเข้ามาแล้วแวะร้านคุณ $C = 2$

ทำให้คิดค่าหวังกำไรได้เป็น

$$P = \left\lfloor \frac{2 \times 6}{6} \right\rfloor - 1 = 2 - 1 = \mathbf{1}$$

ดังนั้นฟังก์ชันจะต้องคืนค่าเป็น 1

ขอบเขตของข้อมูล

- $1 \le N \le 10^9$
- $0 \le K \le \min(N, 2 \times 10^5)$
- $1 \le A[i] \le N$ สำหรับทุก i ที่ $0 \le i \le K-1$ และ $A[i]$ มีค่าที่แตกต่างกันทั้งหมด
- $1 \le Q \le 2 \times 10^5$
- $0 \le W \le 10^9$

ปัญหาย่อย

ปัญหาย่อย	คะแนน	เงื่อนไขเพิ่มเติม
1	6	$N = 50$, $A = [3, 45, 19, 40]$ และมีการเรียก `expected_profit` 3 ครั้ง ($W = 20, 7$ และ 10 ตามลำดับ)
2	8	$N \leq 16$ และ $Q = 1$
3	11	$K \leq 1$
4	13	$K \leq 1000$ และ $Q = 1$
5	15	$Q = 1$
6	16	$K \leq 1000$ และ $Q \leq 1000$
7	18	$W_1 \leq W_2 \leq \cdots \leq W_Q$ (ค่า W ในการเรียก `expected_profit` เรียงจากน้อยไปมาก)
8	13	ไม่มีเงื่อนไขเพิ่มเติม

เกรดเดอร์ตัวอย่าง

ไฟล์เกรดเดอร์จะรับข้อมูลนำเข้าในรูปแบบต่อไปนี้

- บรรทัดที่ 1: รับค่า N K และ Q
- บรรทัดที่ 2: รับค่า $A[0]$ ถึง $A[K-1]$ ตามลำดับ
- บรรทัดที่ 3: รับค่า $W_1, W_2, ..., W_Q$ ตามลำดับ

จากนั้นเกรดเดอร์จะเรียก `init(N, A)` หนึ่งครั้ง และเรียก `expected_profit` จำนวน Q ครั้ง โดยส่งพารามิเตอร์เป็น W ของแต่ละสถานการณ์ และแสดงผลลัพธ์ที่ได้จากการเรียกแต่ละครั้งออกมาทางหน้าจอ