ทริปยาวสุด

ผู้จัดงาน IOI 2023 กำลังเจอปัญหาใหญ่! ผู้จัดลืมวางแผนการเที่ยวเมืองโอพุซตาแซร์ (Ópusztaszer) ในวันที่กำลังจะถึงนี้ แต่มันยังไม่สายเกินไปที่จะวางแผน

ในเมืองโอพุซตาแซร์มีจุดน่าสนใจอยู่ N จุด กำหนดด้วยหมายเลข 0 ถึง $N-1$ บางคู่ของจุดน่าสนใจเหล่านี้มี**ถนน**ที่ใช้เดินทางได้ทั้ง*สองทิศทาง*เชื่อมอยู่ และแต่ละคู่ของจุดน่าสนใจเหล่านี้จะเชื่อมด้วยถนนอย่างมากหนึ่งเส้น ผู้จัดงาน*ไม่ทราบ*ว่าจุดน่าสนใจใดมีถนนเชื่อมอยู่

เราจะเรียกเครือข่ายถนนของเมืองโอพุซตาแซร์ว่ามี **ความหนาแน่น** เป็น **อย่างน้อย** δ ถ้าทุก ๆ จุดน่าสนใจ 3 จุดที่แตกต่างกันมีถนนอย่างน้อย δ เส้นที่เชื่อมจุดน่าสนใจเหล่านั้นอยู่ กล่าวอีกนัยหนึ่งคือ สำหรับแต่ละสามสิ่งอันดับ (triplet) ของจุดน่าสนใจ (u, v, w) ใด ๆ โดยที่ $0 \le u < v < w < N$ นั้น มีคู่อันดับอย่างน้อย δ คู่จาก $(u, v), (v, w)$ และ (u, w) นั้นมีถนนเชื่อมอยู่

ผู้จัด*ทราบ*ค่าจำนวนเต็มบวก D ที่ระบุว่าความหนาแน่นของเครือข่ายถนนมีค่าไม่น้อยกว่า D ให้สังเกตว่าค่า D นั้นไม่สามารถมากกว่า 3 ได้

ผู้จัดงานสามารถ**โทร**หาศูนย์สั่งการทางโทรศัพท์ของเมืองโอพุซตาแซร์เพื่อขอข้อมูลเกี่ยวกับการเชื่อมต่อของถนนระหว่างจุดน่าสนใจต่าง ๆ ตามที่ต้องการได้ การโทรแต่ละครั้งจะต้องระบุอาร์เรย์ไม่ว่างของจุดน่าสนใจจำนวนสองอาร์เรย์คือ $[A[0], \ldots, A[P-1]]$ และ $[B[0], \ldots, B[R-1]]$ โดยที่จุดน่าสนใจเหล่านี้จะต้องแตกต่างกันทั้งหมด กล่าวคือ

- $A[i] \neq A[j]$ สำหรับแต่ละ i และ j ที่ $0 \le i < j < P$
- $B[i] \neq B[j]$ สำหรับแต่ละ i และ j ที่ $0 \le i < j < R$
- $A[i] \neq B[j]$ สำหรับแต่ละ i และ j ที่ $0 \le i < P$ และ $0 \le j < R$

สำหรับการโทรแต่ละครั้ง ศูนย์สั่งการจะรายงานกลับมาว่ามีถนนเชื่อมต่อจุดน่าสนใจใน A กับจุดน่าสนใจใน B หรือไม่ กล่าวโดยละเอียดคือ ศูนย์สั่งการจะไล่ดูทุกคู่ i และ j ใด ๆ ที่ $0 \le i < P$ และ $0 \le j < R$ หากมีสักคู่จุดน่าสนใจ $A[i]$ และ $B[j]$ ใดที่มีถนนเชื่อมอยู่ ศูนย์สั่งการจะคืนค่า `true` แต่ถ้าหากไม่มีเลย ศูนย์สั่งการจะคืนค่า `false`

ให้ **แผนการเที่ยว** ความยาว l คือลำดับของจุดน่าสนใจ $t[0], t[1], \ldots, t[l-1]$ *ที่ไม่ซ้ำกัน* ที่มีถนนเชื่อมระหว่างจุดน่าสนใจ $t[i]$ และ $t[i+1]$ สำหรับทุกค่า i ตั้งแต่ 0 ถึง $l-2$ รวมหัวท้าย แผนการเที่ยวความยาว l จะถูกเรียกว่า**แผนการเที่ยวที่ยาวที่สุด**ถ้าไม่มีแผนการเที่ยวอื่นใดที่มีความยาวอย่างน้อย $l+1$

งานของคุณคือ ช่วยผู้จัดหาแผนการเที่ยวที่ยาวที่สุดในเมืองโอพุซตาแซร์ด้วยการโทรไปยังศูนย์สั่งการ

รายละเอียดการเขียนโปรแกรม

คุณจะต้องเขียนฟังก์ชันต่อไปนี้

```
int[] longest_trip(int N, int D)
```

- N: จำนวนจุดน่าสนใจในเมืองโอพุซตาแซร์
- D: ค่าความหนาแน่นขั้นต่ำที่รับประกันของเครือข่ายถนน
- ฟังก์ชันนี้จะต้องคืนอาร์เรย์ $t = [t[0], t[1], \ldots, t[l-1]]$ ซึ่งระบุถึงแผนการเที่ยวที่ยาวที่สุด
- ฟังก์ชันนี้อาจจะถูกเรียกได้ **หลายครั้ง** ในแต่ละข้อมูลทดสอบ

ฟังก์ชันข้างบนนี้สามารถเรียกใช้ฟังก์ชันต่อไปนี้ได้

```
bool are_connected(int[] A, int[] B)
```

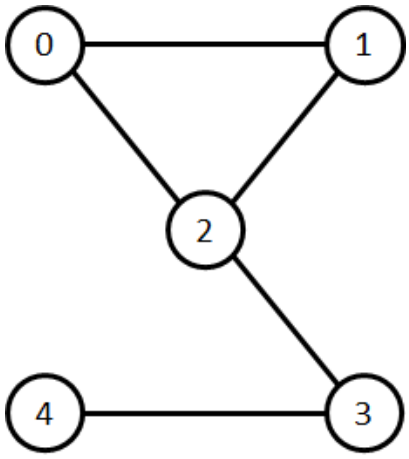
- A: อาร์เรย์ไม่ว่างของจุดน่าสนใจที่แตกต่างกัน
- B: อาร์เรย์ไม่ว่างของจุดน่าสนใจที่แตกต่างกัน
- A และ B จะต้องไม่มีจุดน่าสนใจซ้ำกัน
- ฟังก์ชันนี้จะคืนค่า `true` ถ้ามีถนนเชื่อมจุดน่าสนใจใน A กับจุดน่าสนใจใน B (อย่างน้อยหนึ่งคู่) ถ้าไม่เช่นนั้นจะคืนค่า `false`
- ฟังก์ชันนี้จะถูกเรียกได้ไม่เกิน $32\,640$ ครั้งต่อการเรียก `longest_trip` หนึ่งครั้ง และ รวมทั้งหมดต้องไม่เกิน $150\,000$ ครั้ง
- ความยาวของอาร์เรย์ A และ B ที่ส่งให้ฟังก์ชันนี้ เมื่อรวมทั้งหมดในการเรียกทุกครั้ง จะต้องไม่เกิน $1\,500\,000$

ตัวตรวจนั้น**ไม่ปรับตัว** การส่งคำตอบแต่ละครั้งจะถูกตรวจด้วยข้อมูลทดสอบชุดเดียวกัน กล่าวคือ ค่าของ N และ D รวมถึงคู่ของจุดน่าสนใจที่มีถนนเชื่อมอยู่จะถูกกำหนดไว้คงที่ก่อนที่จะมีการเรียกใช้ `longest_trip` ในแต่ละข้อมูลทดสอบ

ตัวอย่าง

ตัวอย่าง 1

ให้พิจารณาสถานการณ์ที่ $N = 5, D = 1$ และมีถนนแสดงดังรูปต่อไปนี้



ฟังก์ชัน `longest_trip` ถูกเรียกดังต่อไปนี้

```
longest_trip(5, 1)
```

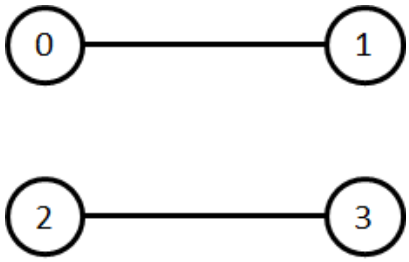
ฟังก์ชันดังกล่าวอาจจะเรียกใช้ `are_connected` ดังต่อไปนี้

การเรียกใช้	คู่ที่มีถนนเชื่อม	ค่าที่คืน
`are_connected([0], [1, 2, 4, 3])`	$(0,1)$ และ $(0,2)$	`true`
`are_connected([2], [0])`	$(2,0)$	`true`
`are_connected([2], [3])`	$(2,3)$	`true`
`are_connected([1, 0], [4, 3])`	ไม่มี	`false`

หลังจากการเรียกครั้งที่สี่ เราได้ผลว่าไม่มีถนนเชื่อมในคู่จุดน่าสนใจ $(1,4)$, $(0,4)$, $(1,3)$ หรือ $(0,3)$ เลย และเพราะว่าความหนาแน่นของถนนมีค่าอย่างน้อยเป็น $D = 1$ เราจะเห็นได้ว่าสำหรับ $(0,3,4)$ นั้น คู่ $(3,4)$ จะต้องมีถนนเชื่อมต่อ และในทำนองเดียวกัน จุดน่าสนใจ 0 และ 1 ก็จะมีถนนเชื่อมต่อด้วยเช่นกัน

ณ จุดนี้ เราสามารถสรุปได้ว่า $t = [1,0,2,3,4]$ เป็นแผนการเที่ยวความยาว 5 และไม่มีแผนการเที่ยวอื่นใดที่มีความยาวมากกว่า 5 อีกแล้ว ดังนั้นฟังก์ชัน `longest_trip` ควรต้องคืนค่า $[1,0,2,3,4]$

พิจารณาอีกสถานการณ์หนึ่งที่ $N = 4$, $D = 1$ และมีถนนแสดงดังรูปต่อไปนี้



ฟังก์ชัน `longest_trip` ถูกเรียกดังต่อไปนี้

```
longest_trip(4, 1)
```

ในกรณีนี้ ความยาวของแผนการเที่ยวที่ยาวที่สุดคือ 2 ดังนั้น หลังจากการเรียกฟังก์ชัน `are_connected` ไม่กี่ครั้ง ฟังก์ชัน `longest_trip` จะสามารถคืนค่าใด ๆ ต่อไปนี้ก็ได้ $[0,1]$, $[1,0]$, $[2,3]$ หรือ $[3,2]$

ตัวอย่าง 2

ปัญหาย่อย 0 มีข้อมูลทดสอบที่เป็นตัวอย่างเพิ่มเติมที่มีจุดน่าสนใจ $N = 256$ จุด ข้อมูลทดสอบนี้ถูกรวมอยู่ในชุดไฟล์แนบที่คุณสามารถดาวน์โหลดได้จากระบบแข่งขัน

ข้อจำกัด

- $3 \leq N \leq 256$

- ผลรวมของ N จากทุกการเรียก `longest_trip` จะมีค่าไม่เกิน $1\,024$ ในแต่ละข้อมูลทดสอบ
- $1 \le D \le 3$

ปัญหาย่อย

1. (5 คะแนน) $D = 3$
2. (10 คะแนน) $D = 2$
3. (25 คะแนน) $D = 1$ ให้ $l^\star$ ระบุถึงความยาวของแผนการเที่ยวที่ยาวที่สุด ฟังก์ชัน `longest_trip` ไม่จำเป็นต้องคืนค่าแผนการเที่ยวที่ยาว $l^\star$ แต่สามารถคืนค่าแผนการเที่ยวที่มีความยาวอย่างน้อย $\left\lceil \frac{l^\star}{2} \right\rceil$ แทนได้
4. (60 คะแนน) $D = 1$

ในปัญหาย่อยที่ 4 คะแนนของคุณจะถูกคิดโดยพิจารณาจากจำนวนครั้งที่เรียกใช้ฟังก์ชัน `are_connected` ต่อการเรียก `longest_trip` หนึ่งครั้ง ให้ q คือ จำนวนครั้งในการเรียก `are_connected` ที่มากที่สุดจากการเรียก `longest_trip` ของข้อมูลทดสอบทั้งหมดในปัญหาย่อยนั้น คะแนนของคุณในปัญหาย่อยนั้นจะเป็นดังตารางต่อไปนี้

เงื่อนไข	คะแนน
$2\,750 < q \le 32\,640$	20
$550 < q \le 2\,750$	30
$400 < q \le 550$	45
$q \le 400$	60

หากว่าการเรียกฟังก์ชัน `are_connected` ในข้อมูลทดสอบใด ๆ ไม่ตรงกับข้อกำหนดที่ระบุไว้ในรายละเอียดการเขียนโปรแกรม หรือหากว่าอาร์เรย์ที่คืนมาจาก `longest_trip` นั้นไม่ถูกต้อง คะแนนของคำตอบของคุณในปัญหาย่อยนั้นจะเป็น 0

เกรดเดอร์ตัวอย่าง

ให้ C คือจำนวนสถานการณ์ เป็นจำนวนสถานการณ์ หรือก็คือ จำนวนครั้งที่จะเรียกฟังก์ชัน `longest_trip` เกรดเดอร์ตัวอย่างอ่านข้อมูลนำเข้าตามรูปแบบต่อไปนี้:

- บรรทัดที่ 1: C

จากนั้นต่อด้วยคำบรรยายของทั้ง C สถานการณ์

เกรดเดอร์ตัวอย่างอ่านคำบรรยายสำหรับแต่ละสถานการณ์ตามรูปแบบต่อไปนี้:

- บรรทัดที่ 1: N D
- บรรทัดที่ $1 + i$ $(1 \le i < N)$: $U_i[0]$ $U_i[1]$ $\ldots$ $U_i[i-1]$

แต่ละ U_i $(1 \le i < N)$ คืออาร์เรย์ขนาด i ซึ่งอธิบายถึงคู่จุดน่าสนใจที่มีถนนเชื่อม สำหรับแต่ละ i และ j โดยที่ $1 \le i < N$ และ $0 \le j < i$

- ถ้าจุดน่าสนใจ j และ i มีถนนเชื่อม ค่าของ $U_i[j]$ จะเป็น 1
- ถ้าไม่มีถนนเชื่อมคู่จุดน่าสนใจ j และ i ค่าของ $U_i[j]$ จะเป็น 0

ในแต่ละกรณี ก่อนที่จะมีการเรียก `longest_trip` เกรดเดอร์ตัวอย่างจะตรวจสอบว่าค่าความหนาแน่นของเครือข่ายถนนมีค่าอย่างน้อย D หรือไม่ หากไม่เป็นจริง เกรดเดอร์ตัวอย่างจะพิมพ์ข้อความ `Insufficient Density` แล้วหยุดทำงาน

หากเกรดเดอร์ตัวอย่างตรวจพบการเรียกใช้งานฟังก์ชันที่ผิดพลาด ไม่ตรงกับข้อกำหนด เกรดเดอร์ตัวอย่างจะพิมพ์ข้อความ `Protocol Violation: <MSG>` โดยที่ `<MSG>` คือข้อความที่ระบุความผิดพลาดดังต่อไปนี้:

- `invalid array`: ในการเรียก `are_connected` มีอาร์เรย์ A หรือ B อย่างน้อยหนึ่งอาร์เรย์ที่
  - เป็นอาร์เรย์ว่าง หรือ
  - มีข้อมูลที่ไม่ใช่จำนวนเต็มที่มีค่าตั้งแต่ 0 ถึง $N - 1$ รวมหัวท้าย หรือ
  - มีข้อมูลอย่างน้อยสองตัวซ้ำกัน
- `non-disjoint arrays`: ในการเรียก `are_connected` นั้น อาร์เรย์ A และ B มีข้อมูลบางตัวซ้ำกัน
- `too many calls`: จำนวนครั้งในการเรียก `are_connected` เกิน $32\,640$ จากการเรียก `longest trip` ครั้งปัจจุบัน หรือ เกิน $150\,000$ จากการเรียกทั้งหมด
- `too many elements`: จำนวนจุดน่าสนใจที่ส่งให้ `are_connected` รวมทุกครั้ง เกิน $1\,500\,000$

ในกรณีที่ไม่พบความผิดพลาดใด ๆ ให้ $t[0], t[1], \ldots, t[l-1]$ สำหรับค่า l ที่ไม่เป็นลบคืออาร์เรย์ที่คืนมาจาก `longest_trip` เกรดเดอร์ตัวอย่างจะพิมพ์ข้อมูลจำนวนสามบรรทัดดังต่อไปนี้:

- บรรทัดที่ 1: l
- บรรทัดที่ 2: $t[0]$ $t[1]$ $\ldots$ $t[l-1]$
- บรรทัดที่ 3: จำนวนครั้งที่เรียก `are_connected` ในสถานการณ์นี้

สุดท้าย เกรดเดอร์ตัวอย่างจะพิมพ์:

- บรรทัดที่ $1 + 3 \cdot C$: จำนวนครั้งในการเรียก `are_connected` ที่มากที่สุดจากการเรียก `longest_trip` ทั้งหมด