ตาบอดเดินกรุงเทพ

ชายคนหนึ่งตาบอด เขามองเห็นไม่ชัดถึงขั้นที่ว่ามองไม่เห็นถนน วันหนึ่งเขาเดินเข้ากรุงเทพ ชายตาบอดอยากใช้ศักยภาพทั้งหมดที่มีของเขาในการสร้างแผนที่กรุงเทพขึ้นมา เคยมีคนสงสัยว่าเวลาชายตาบอดเดินทางไปเยี่ยมผู้คนตามต่างจังหวัด เขาเห็นดเหนื่อยบ้างหรือไม่ แต่ชาวตาบอดก็ยืนกรานไม่เห็นดเหนื่อย ตั้งมั่นเดินทางไปครบทุกจังหวัด ถึงเวลาแล้วที่เขาจะเดินทางทั่วกรุงเทพ

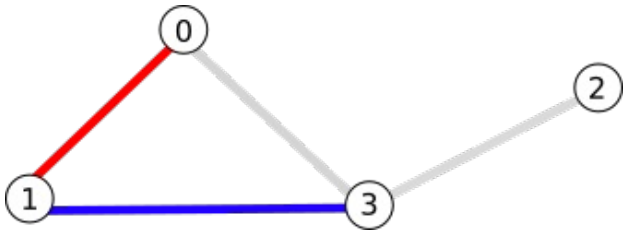
กรุงเทพเป็นเมืองที่มีทางเดินช่วยเหลือคนตาบอด (Braille Block) มีทางเดินสุดพิลึกกึกกือ นอกจากการเดินทางบนถนนแล้วยังมีการเดินทางบนรถไฟฟ้า BTS กับรถไฟใต้ดิน MRT ที่อำนวยความสะดวกไว้สำหรับคนตาบอด

เราสามารถพิจารณากรุงเทพเป็นกราฟ ประกอบด้วยจุดยอด N จุด ($50 \leq N \leq 520$) เรียกเป็นหมายเลข 0 ถึง $N-1$ โดยแต่ละคู่ของจุดสามารถเชื่อมกันได้ 3 รูปแบบคือ (1) ด้วยทางเท้า (2) ด้วยรถไฟฟ้า BTS หรือ (3) ด้วยรถไฟใต้ดิน MRT

รับประกันว่าจุดยอดสองจุดใดๆ จะมีการเชื่อมต่อกัน*โดยตรง*ได้ไม่เกินแบบเดียว (กล่าวคือไม่มีจุดยอด s, t ที่ มีทั้งรถไฟฟ้าและทางเดินเชื่อม s กับ t) และไม่มีจุดไหนมีเส้นเชื่อมเชื่อมตัวเอง นอกจากนี้รับประกันว่าจากจุดยอดใด ๆ จะสามารถเดินทางไปหาจุดยอดอื่นใด ๆ ได้ผ่านการเชื่อมต่อที่มีนี้

ชายตาบอดสามารถเดินทางไปทั่วเมืองได้เนื่องจากเขาไม่รู้จักเหน็ดเหนื่อย คุณสามารถขอให้เขาเดินจากจุดยอด s ไปยังจุดยอด t ได้ แล้วเขาจะตอบว่าใช้เวลาน้อยที่สุดในการเดินทางเท่าใด การเดินทางเท้า ไม่ว่าทางเท้าใดก็ตาม จะใช้เวลา $2\,596\,418\,101$ ไมโครวินาทีพอดีเสมอ การใช้รถไฟฟ้า BTS เดินทางระหว่างสถานี จะใช้เวลา $840\,634\,349$ ไมโครวินาทีพอดีเสมอ และการใช้รถไฟใต้ดิน MRT เดินทางระหว่างสถานี จะใช้เวลา $590\,846\,489$ ไมโครวินาทีพอดีเสมอ

พิจารณาตัวอย่างแผนที่กรุงเทพดังนี้ (เส้นสีเทาคือทางเท้า, เส้นสีแดงคือ BTS, และเส้นสีน้ำเงินคือ MRT)



ชายตาบอดจะบอกเวลาตามคำถามต่าง ๆ ดังนี้ (ฟังก์ชัน `ask` จะตรงกับ api ที่คุณจะต้องเขียน)

คำถาม	s	t	ค่าคืนกลับ	หมายเหตุ
`ask(0,1)`	0	1	840,634,349	ไปทาง BTS โดยตรง
`ask(3,0)`	3	0	1,431,480,838	เดินทางด้วย MRT + BTS เร็วกว่าเดินทางตรงด้วยทางเท้า
`ask(2,3)`	2	3	2,596,418,101	เดินทางเท้า
`ask(2,0)`	2	0	4,027,898,939	เดินทางเท้า ต่อ MRT ต่อ BTS
`ask(1,3)`	1	3	590,846,489	เดินทางด้วย MRT

ให้คุณสร้างกราฟกรุงเทพ กราฟที่คุณสร้างอาจจะไม่จำเป็นต้องเหมือนกราฟตั้งต้นทุกอย่าง เพียงแต่ต้องรับประกันว่าเมื่อพิจารณาจากคู่ของจุดยอดใด ๆ เวลาในการเดินทางที่น้อยที่สุดจะต้องเท่ากับกับกราฟกรุงเทพจริง ๆ เสมอ

คุณไม่ต้องการรบกวนชายตาบอดมากนัก ดังนั้น ยิ่งถามน้อยครั้งก็ยิ่งดี (อย่าลืมพิจารณาปัญหาย่อย และเกณฑ์การให้คะแนน)

รายละเอียดการเขียนโปรแกรม

คุณจะต้องเขียนฟังก์ชันต่อไปนี้

```
vector<vector<int>> build_graph(int N)
```

- ฟังก์ชันนี้จะถูกเรียกหนึ่งครั้ง โดยมีการระบุจำนวนจุยอด (N)
- ฟังก์ชันจะต้องคืนกราฟกรุงเทพที่สร้างได้เป็นเวกเตอร์ ขนาดของเวกเตอร์เท่ากับจำนวนเส้นเชื่อมของกราฟ ข้อมูลแต่ละตัวในเวกเตอร์จะระบุข้อมูลการเชื่อมต่อระหว่างคู่ของจุดยอด สมมติว่า x เป็นเวกเตอร์แทนการเชื่อมต่อหนึ่งอัน เราจะต้องให้ $x[0]$ และ $x[1]$ แทนหมายเลขจุดยอด และ $x[2]$ มีค่าเป็น 1, 2, หรือ 3 ขึ้นกับประเภทการเชื่อมต่อ

ฟังก์ชันดังกล่าวสามารถเรียกฟังก์ชันต่อไปนี้เพื่อให้ชายตาบอดเดินทาง

```
long long ask(int s, int t)
```

- เป็นการเรียกให้ชายตาบอดเดินทางระหว่างจุดยอด s และ t
- ฟังก์ชันจะคือเวลาบนเส้นทางที่สั้นที่สุดจากจุดยอด s ไปยังจุดยอด t

การให้คะแนน

ถ้าคุณสร้างกราฟที่เวลาในการเดินทางระหว่างทุกแยก ตรงกับเวลาของกรุงเทพครบทุกกรณีทดสอบในปัญหาย่อย คุณจะได้คะแนนในปัญหาย่อยนั้น ในการให้คะแนนนั้น คะแนนที่คุณได้จะขึ้นกับจำนวนครั้งที่คุณให้ชายตาบอดเดิน เปรียบเทียบระหว่างโปรแกรมของคุณและโปรแกรมของคนอื่น ๆ ที่ตอบถูกทุกกรณีทดสอบในปัญหาย่อยนั้นรวมทั้งโปรแกรมของกรรมการด้วย (ในวันสอบจริง จะมีประกาศจำนวนการเรียกถามของโปรแกรมกรรมการล่วงหน้า จำนวนการเรียกถามของโปรแกรมของกรรมการไม่จำเป็นจะต้อง

เป็นจำนวนที่ดีที่สุดที่สามารถทำได้)

สำหรับแต่ละกรณีทดสอบ สมมติว่าคุณถาม Q ครั้ง และจำนวนการถามที่น้อยที่สุดคือ R ครั้ง ถ้า $Q \le 1.03 \cdot R$ คุณจะได้คะแนนเต็ม (นั่นคืออยู่ภายใน 3% ของการถามที่ดีที่สุด) ถ้าไม่เช่นนั้น คะแนนที่คุณได้จะเป็นสัดส่วน $(Q/R)^{1.5}$ ของคะแนนเต็ม คะแนนที่คุณได้ของปัญหาย่อยจะเท่ากับคะแนนน้อยสุดที่คุณได้ คุณจะทราบคะแนนที่ได้จริง ๆ หลังการแข่งขัน

ระหว่างการแข่ง ถ้าโปรแกรมคุณทำงานถูกต้อง เกรดเดอร์จะแจ้งจำนวนการถามของคุณในแต่ละกรณีทดสอบในรูปแบบ `(number)` ในกรณีที่คุณทำงานผิดพลาด เกรดเดอร์จะแจ้งดังนี้: `L` ถามเกิน $250\,000$ ครั้ง, `G` กราฟมีข้อมูลผิดพลาด เช่น ขอบเขตจุดยอดผิด ประเภทการเชื่อมต่อผิด, `D` ระยะทางที่ได้ไม่ตรง

หมายเหตุ: คะแนนที่คุณได้คือคะแนนรวมของคะแนนที่ดีที่สุดจากที่คุณส่งของแต่ละปัญหาย่อยที่คุณทำได้ (เราจะคิดแบบนี้กับทุกข้อ แต่ขอประกาศไว้ตรงนี้ด้วย เพื่อความชัดเจน)

ขอบเขต

- $50 \le N \le 520$
- ถามได้ไม่เกิน $250\,000$ ครั้ง

ปัญหาย่อย

1. (4 คะแนน) กรุงเทพมีการเชื่อมต่อรูปแบบเดียวเท่านั้น (อาจจะเป็นทางเท้า, BTS, หรือ MRT) และกราฟจะเป็นกราฟแบบใดก็ได้
2. (8 คะแนน) กรุงเทพมีการเชื่อมต่อรูปแบบเดียวเท่านั้น (อาจจะเป็นทางเท้า, BTS, หรือ MRT) และกราฟเป็นเส้นทาง (path) นั่นคือในกราฟทุก ๆ จุดยอดมีจุดยอดติดกัน 2 จุดยอดเสมอ ยกเว้นจุดยอดปลายสองจุดยอด ที่จะมีจุดยอดที่ติดกับมันแค่ 1 จุด
3. (9 คะแนน) กรุงเทพมีการเชื่อมต่อรูปแบบเดียวเท่านั้น (อาจจะเป็นทางเท้า, BTS, หรือ MRT) และกราฟเป็นวงรอบ (cycle) นั่นคือในกราฟทุก ๆ จุดยอดมีจุดยอดติดกัน 2 จุดยอดเสมอ
4. (10 คะแนน) กรุงเทพมีการเชื่อมต่อรูปแบบเดียวเท่านั้น (อาจจะเป็นทางเท้า, BTS, หรือ MRT) และกราฟจะเป็นเส้นทาง (path) หรือ วงรอบ (cycle)
5. (14 คะแนน) กรุงเทพมีการเชื่อมต่อรูปแบบเดียวเท่านั้น (อาจจะเป็นทางเท้า, BTS, หรือ MRT) และกราฟเป็นต้นไม้ไบนารีแบบสมบูรณ์ (complete binary tree) ที่มีจำนวนจุดยอดเท่ากับ $2^k - 1$ สำหรับบางจำนวนเต็ม k
6. (4 คะแนน) กราฟจะเป็นกราฟแบบใดก็ได้
7. (11 คะแนน) กราฟเป็นเส้นทาง (path) นั่นคือในกราฟทุก ๆ จุดยอดมีจุดยอดติดกัน 2 จุดยอดเสมอ ยกเว้นจุดยอดปลายสองจุดยอด ที่จะมีจุดยอดที่ติดกับมันแค่ 1 จุด
8. (12 คะแนน) กราฟเป็นวงรอบ (cycle) นั่นคือในกราฟทุก ๆ จุดยอดมีจุดยอดติดกัน 2 จุดยอดเสมอ
9. (13 คะแนน) กราฟจะเป็นเส้นทาง (path) หรือ วงรอบ (cycle)
10. (15 คะแนน) กราฟเป็นต้นไม้ไบนารีแบบสมบูรณ์ (complete binary tree) ที่มีจำนวนจุดยอดเท่ากับ $2^k - 1$ สำหรับบางจำนวนเต็ม k

ตัวอย่าง

จากตัวอย่างข้างต้น จะเรียกฟังก์ชัน `build_graph` ดังนี้

```
build_graph(4)
```

ฟังก์ชันดังกล่าวอาจจะเรียกฟังก์ชัน `ask` ได้หลายแบบ ผลลัพธ์ที่คืนกลับมาจะเป็นดังตารางในตัวอย่างข้างต้น

ฟังก์ชัน `build_graph` จะต้องคืนกราฟที่สร้าง โดยอาจจะคืนเป็น

```
[[0, 1, 2],
 [0, 3, 1],
 [1, 3, 3],
 [2, 3, 1]]
```

หรือ ดังด้านล่าง

```
[[0, 1, 2],
 [1, 3, 3],
 [2, 3, 1]]
```

หรือจะเป็นรูปแบบอื่น หรือเรียงเส้นเชื่อมแบบอื่นที่เวลาที่ใช้ในการเดินทางระหว่างทุกจุดยอดเหมือนกับในรูปด้านบนก็ถูกต้องทั้งหมด

เกรดเดอร์ตัวอย่าง

เกรดเดอร์ตัวอย่างจะใช้เวลาการทำงานมากกว่าเกรดเดอร์จริงเนื่องจากมีการคำนวณเวลาในการเดินทางก่อนการทำงาน และเพื่อการตรวจ

เกรดเดอร์ตัวอย่างอ่านข้อมูลดังนี้

- บรรทัดที่ 1: N, M (จำนวนจุดยอด และจำนวนเส้นเชื่อม)
- บรรทัดที่ $2, \ldots, M+1$: u, v และ t (เพื่อระบุว่ามีการเชื่อมระหว่างจุดยอด u และ v ด้วยรูปแบบ t โดย $t = 1$ ถ้าเป็นทางเท้า, $t = 2$ ถ้าเป็น BTS, และ $t = 3$ ถ้าเป็น MRT)

เกรดเดอร์จะพิมพ์จำนวนครั้งที่ถาม พิมพ์กราฟที่ตอบ นอกจากนี้ในเกรดเดอร์สำหรับผู้เข้าแข่งขัน เกรดเดอร์จะพิมพ์ว่าผลลัพธ์ถูกต้องหรือไม่ด้วย หรือถ้าผิดพลาดจะบอกคู่ที่เวลาในการเดินทางผิดพลาด

ข้อจำกัด

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