Navigation

تتميز تايوان بكثافة عالية من الجبال الشاهقة, بحيث تضم أكثر من 250 قمة فوق 3,000 متر. يخطط آ-مينغ لإنشاء شبكات توصيل آلية باستخدام الحبال الانزلاقية في سلسلة الجبال الوسطى في تايوان، لنقل الطرود بين القرى النائية.

حسب خطة آ-مينغ, كل شبكة تتكون من N رأس مرقمة من 0 الى $N-1$. بين هاته الرؤوس, **بالضبط** $K=6$ منها هي محطات طاقة و $N-K$ الأخرى كلها رؤوس قرى. الرؤوس متصلة من خلال $N-1$كابلات **ثنائية الاتجاه**, مرقمة من 0 الى $N-2$. من أجل كل i $(0 \le i \le N-2)$, الكابل i يربط الرأسين $U[i]$ و $V[i]$. كل كابل يربط رأسين مختلفين, وكل زوج من الرؤوس متصل عن طريق كابل واحد على الأكثر. من المضمون أنه يمكن الانتقال بين أي ثنائية من الرؤوس باستعمال الكابلات.

المسافة بين رأسين a و b يرمز لها ب$d(a,b)$ ومعرفة كالتالي:

- اذا كان $a=b$ فان $d(a,b)=0$.
- والا, $d(a,b)$هو العدد الأدنى من الكابلات التي نحتاجها للانتقال من a الى b.

نقول أن **معامل التفرع** الخاص بالشبكة يساوي δ على الأقل اذا كان كل رأس من الشبكة اما متصلا بكابل واحد بالضبط أو على الأقل δ كابل. آ-مينغ تعرف عدد صحيح موجب B بحيث معامل التفرع للشبكة هو على الأقل B.

آ-مينغ قامت بتحضير 100 نوع مختلف من الكابلات, مرقمة $0, 1, \cdots, 99$. كل كابل في الشبكة يتم بناؤه من خلال أحد هاته الأنواع.

ستتحرك الروبوتات على طول كابلات الحبال الانزلاقية لتنفيذ مهام التوصيل. يرغب آ-مينغ في تثبيت برنامج ملاحة يساعد الروبوتات على العودة إلى محطات الطاقة وشحن نفسها. ومع ذلك، فإن الروبوتات تمتلك ذاكرة منخفضة للغاية. لذلك، عند تصميم برنامج الملاحة، تتنقل الروبوتات بناءً فقط على عدد محطات الطاقة $K=6$، ومعامل التفرع المضمون B، وأنواع الكابلات المتصلة بالموقع الحالي للروبوت.

عندما يرغب الروبوت في التنقل عند رأس قرية u متصل بكابلين أو أكثر، يقوم الروبوت أولاً بمسح الكابلات المتصلة بالرأس u بترتيب عشوائي. ثم يتم تزويد البرنامج بقائمة تحتوي على أنواع الكابلات بهذا الترتيب. بالنسبة لكل كابل، يحتاج البرنامج إلى حساب عدد محطات الطاقة بحيث يؤدي اتباع ذلك الكابل إلى تقليل المسافة بين الروبوت والمحطة.

على وجه التحديد، لنفترض أن $v_1, \ldots, v_k$ (حيث $k \ge 2$) هي الرؤوس المتصلة مباشرة بالرأس u بواسطة كابل، وأن c_i (حيث $1 \le i \le k$) هو نوع الكابل الذي يربط الرأس u بالرأس v_i. يتم بعد ذلك تزويد البرنامج بالقيم K و B والقائمة $c_1, c_2, \ldots, c_k$. من أجل كل i $(1 \le i \le k)$، يجب أن يحدد البرنامج عدد محطات الطاقة p التي تحقق $d(v_i, p) < d(u, p)$.

مهمتك هي ابتكار استراتيجية لتعيين أنواع الكابلات وتصميم برنامج الملاحة. تعتمد نتيجة حلك (النقاط) على عدد الأنواع المختلفة من الكابلات المستخدمة (انظر قسم المهام الفرعية والنقاط للحصول على التفاصيل).

تفاصيل التنفيذ

تحتاج إلى تنفيذ دالتين (إجراءين)، إحداهما لتعيين أنواع الكابلات والأخرى لبرنامج الروبوتات.

الدالة التي يجب عليك تنفيذها **لتعيين أنواع الكابلات** هي:

```
std::vector construct_network(int N, int K, int B,
                                  std::vector U, std::vector V,
                                  std::vector P)
```

- N: عدد الرؤوس.
- K: عدد محطات الطاقة.
- B: الحد الأدنى المضمون لمعامل التفرع في الشبكة.
- U, V: مصفوفتان بطول $N - 1$ تصفان اتصالات الكابلات.
- P: مصفوفة بطول $K = 6$ تصف رؤوس محطات الطاقة.
- يتم استدعاء هذه الدالة **10,000 مرة على الأكثر** لكل حالة اختبار.

يجب أن تُرجع هذه الدالة مصفوفة T بطول $N - 1$:

- يتم استخدام كابل من النوع $T[i]$ لربط الرأسين $U[i]$ و $V[i]$.
- يجب أن يحقق كل عنصر $T[i]$ الشرط $0 \leq T[i] < 100$.

الدالة التي يجب عليك تنفيذها **لبرنامج الروبوتات** هي:

```
std::vector navigate(int K, int B, std::vector C)
```

- K: عدد محطات الطاقة.
- B: الحد الأدنى المضمون لمعامل التفرع في الشبكة.
- C: قائمة بأنواع جميع الكابلات المتصلة بـ **رأس قرية متصل بكابلين على الأقل**، بترتيب عشوائي.
- من المضمون أن طول المصفوفة C هو B على الأقل.
- يتم استدعاء هذه الدالة 100,000 مرة على الأكثر لكل حالة اختبار.
- قد لا تعتمد الدالة `Maps` على بنية الشبكة الأصلية؛ على وجه الخصوص، قد يقوم نظام التقييم بإعادة ترتيب جميع استدعاءات `Maps` عبر الشبكات المختلفة المبنية ضمن نفس حالة الاختبار.

يجب أن تُرجع هذه الدالة مصفوفة D:

- يجب أن يكون طول المصفوفة D مساويًا لطول المصفوفة C.
- يجب أن يكون $D[i]$ هو عدد محطات الطاقة التي يؤدي اتباع الكابل المقابل لـ $C[i]$ إلى تقليل المسافة بين الروبوت والمحطة.

لاحظ أنه في نظام التقييم، يتم استدعاء الدالتين `construct_network` و `Maps` **في عمليتين منفصلتين.**

القيود

- $K = 6$.
- $K < N \leq 100,000$.
- مجموع N على جميع استدعاءات `construct_network` لا يتجاوز 100,000 في كل حالة اختبار.
- $0 \leq U[i] < V[i] < N$ لكل $0 \leq i \leq N-2$.
- من الممكن الانتقال من أي رأس إلى أي رأس آخر باستخدام الكابلات.
- $0 \leq P[0] < P[1] < \cdots < P[K-1] < N$.
- كل مصفوفة C هي تبديلة لقائمة أنواع جميع الكابلات المتصلة برأس قرية متصل بكابلين على الأقل.
- مجموع أطوال المصفوفات C على جميع استدعاءات `Maps` لا يتجاوز 200,000 في كل حالة اختبار.

المهام الفرعية والنقاط

المهمة الفرعية	النقاط	قيود إضافية
1	6	$B = 2, N = 7$.
2	14	$B = 2, U[i] = i, V[i] = i+1$ بحيث i لكل $0 \leq i < N-1$.
3	20	$B = 5$.
4	20	$B = 4$.
5	40	$B = 2$.

في أي من حالات الاختبار، إذا تحقق على الأقل أحد الشروط التالية، فإن نقطة حلك لتلك الحالة ستكون 0 (سيتم الإبلاغ عنها كـ `Output isn't correct` في نظام التقييم CMS):

- القيمة المرجعة لأي استدعاء للدالة `construct_network` غير صالحة.
- القيمة المرجعة لأي استدعاء للدالة `Maps` غير صحيحة.

عدا ذلك، ليكن S هو أصغر عدد صحيح أكبر من جميع الأعداد في كل مصفوفة T يتم إرجاعها من كل استدعاء للدالة `construct_network`.

بعبارة أخرى، S هو أصغر عدد صحيح بحيث جميع الشبكات المبنية تستخدم فقط كابلات من النوع $0, 1, \ldots, S-1$.

نقاطك في كل مهمة فرعية تعتمد على S، كالتالي:

الشرط	المهمة الفرعية 1	المهمة الفرعية 2	المهمة الفرعية 3 و 4	المهمة الفرعية 5
$100 < S$	0	0	0	0
$16 \leq S \leq 100$	2	2	$12 - \log_2 S$	$24 - 2\log_2 S$
$10 \leq S \leq 15$	2	4	$16 - 0.5S$	$32 - S$
$7 \leq S \leq 9$	2	6	$21 - S$	$42 - 2S$
$S = 6$	2	10	15	30
$S = 5$	**6**	**14**	17.5	**40**
$S \leq 4$	**6**	**14**	**20**	**40**

على وجه الخصوص، في المهام الفرعية 1 و 2 و 5، ستحصل على العلامة الكاملة إذا كان $S \leq 5$، وفي المهام الفرعية 3 و 4، ستحصل على العلامة الكاملة إذا كان $S \leq 4$.

مثال

لنفترض السيناريو الذي يكون فيه $N = 9$ و $K = 6$ و $B = 2$.

مخطط الشبكة محدد بواسطة $U = [0, 0, 0, 2, 2, 2, 2, 7]$ و $V = [1, 2, 3, 4, 5, 6, 7, 8]$.

محطات الطاقة هي $P = [1, 3, 4, 5, 6, 7]$.

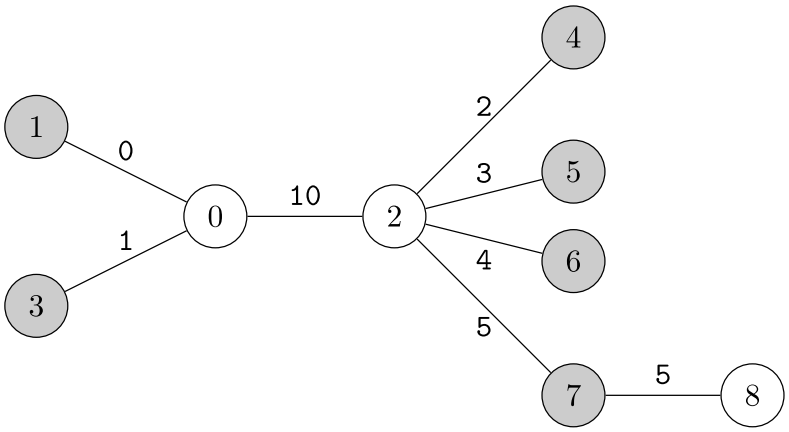
في هذا السيناريو، يتم الاستدعاء التالي للعملية الأولى:

```
construct_network(9, 6, 2, [0, 0, 0, 2, 2, 2, 2, 7],
                           [1, 2, 3, 4, 5, 6, 7, 8],
                           [1, 3, 4, 5, 6, 7])
```

لنفترض أن آ-مينغ قام ببناء الشبكة عن طريق إرجاع:

```
[0, 10, 1, 2, 3, 4, 5, 5]
```

الشبكة المبنية موضحة كالتالي.



ثم، في العملية الثانية، لنفترض أن الروبوت يحتاج إلى التنقل عند الرأس 0.

الرأس 0 متصل بالرؤوس 1 و 2 و 3.

إذا قرأ الروبوت نوع الكابل بهذا الترتيب، فسيتم إجراء الاستدعاء التالي:

```
navigate(6, 2, [0, 10, 1])
```

بناءً على بنية الشبكة:

- المحطة الأولى للطاقة، الرأس 1، لها أقصر مسافة إلى نفسها.

  على وجه الخصوص، $0 = d(1,1) < 1 = d(0,1) < 2 = d(2,1) = d(3,1)$.

- المحطة الثانية للطاقة، الرأس 3، لها أقصر مسافة إلى نفسها.

  على وجه الخصوص، $0 = d(3,3) < 1 = d(0,3) < 2 = d(1,3) = d(2,3)$.

- محطات الطاقة الأخرى، الرؤوس 4 و 5 و 6 و 7، لها مسافات أقصر إلى الرأس 2.

  على وجه الخصوص، $1 = d(2,u) < 2 = d(0,u) < 3 = d(1,u) = d(3,u)$ عندما يكون $u = 4, 5, 6,$ أو 7.

عدد محطات الطاقة التي تتناقص المسافة إليها بعد اتباع الكابلات $(0,1)$ و $(0,2)$ و $(0,3)$ هو 1 و 4 و 1 على التوالي.

وبالتالي، يجب أن تُرجع الدالة:

```
[1, 4, 1]
```

لاحظ أنه يمكن أيضًا استدعاء `Maps` لتبديلات مختلفة من المصفوفة C.

على سبيل المثال، `Maps(6, 2, [1, 0, 10])` هو أيضًا استدعاء محتمل للرأس 0.

يجب أن تُرجع الدالة `[1, 1, 4]` بناءً على ذلك.

لنفترض أن الروبوت يحتاج إلى التنقل عند الرأس 2.

يتم إجراء الاستدعاء التالي:

```
navigate(6, 2, [10, 2, 3, 4, 5])
```

يجب أن تُرجع الدالة:

```
[2, 1, 1, 1, 1]
```

في هذه الشبكة، **لن** يتم استدعاء الدالة `Maps` للرؤوس الأخرى أبدًا، لأن:

- الرؤوس 1 و 3 و 4 و 5 و 6 و 7 هي جميعها محطات طاقة، و
- الرأس 8 متصل بكابل واحد فقط.

في حالة الاختبار هذه، أنواع الكابلات المستخدمة هي 0 و 1 و 2 و 3 و 4 و 5 و 10.

وبالتالي، سيتم استخدام $S = 11$ لتحديد النقاط.

تحتوي حزمة المرفقات لهذه المهمة على عينتين أخريين للمدخلات والمخرجات لبرنامج التقييم التجريبي بالإضافة إلى هذا المثال.

برنامج التقييم التجريبي (Sample Grader)

يقوم برنامج التقييم التجريبي باستدعاء كل من `construct_network` و `Maps` في نفس العملية (process).

بالإضافة إلى ذلك، عندما يستدعي برنامج التقييم التجريبي `Maps`، يتم سرد أنواع الكابلات بنفس الترتيب الذي تظهر به الكابلات المقابلة لها في المدخلات.

كلا السلوكين يختلفان عن برنامج التقييم الفعلي.

صيغة المدخلات:

```
T
(scenario 0)
(scenario 1)
...
(scenario T-1)
```

كل سيناريو يأتي بالصيغة التالية:

```
N
B
U[0] V[0]
U[1] V[1]
...
U[N-2] V[N-2]
K
P[0] P[1] ... P[K-1]
Q
X[0] X[1] ... X[Q-1]
```

في كل سيناريو، سيقوم برنامج التقييم التجريبي باستدعاء `Maps` بعدد Q مرة، الاستدعاء رقم i سيكون بمعلومات الرأس $X[i-1]$.

لاحظ أن برنامج التقييم التجريبي يدعم تعيين K لأعداد صحيحة أخرى غير 6.

صيغة المخرجات:

إذا كانت أي من المصفوفات المرجعة من `construct_network` أو `Maps` غير صالحة، فإن برنامج التقييم التجريبي يتوقف ويطبع السبب.

عدا ذلك، يطبع برنامج التقييم التجريبي كل مصفوفة D مُرجعة من استدعاء `Maps` في سطر مستقل.

بعد معالجة جميع السيناريوهات، يطبع برنامج التقييم التجريبي سطرًا إلى المخرج القياسي للأخطاء (standard error):

```
S = S'
```

حيث S' هو أصغر عدد صحيح أكبر من جميع الأعداد في كل مصفوفة T تم إرجاعها من كل استدعاء للدالة `construct_network` في حالة الاختبار.