Eng uzun sayohat

IOI 2023 tashkilotchilari katta muammoga duch kelishmoqda! Ular kelgusi kun uchun Opuztazerga sayohatni rejalashtirishni unutishdi. Ammo, ehtimol, hali kech emas ...

Opuztazerda 0 dan N-1 gacha indekslangan N ta diqqatga sazovor joylar bor. Bu diqqatga sazovor joylarning ba'zi juftliklari o'zaro *ikki tomonlama* **yo'llar** orqali bog'langan. Har bir diqqatga sazovor joylar juftligi ko'pi bilan bitta yo'l orqali bog'langan. Tashkilotchilar qaysi diqqatga sazovor joylar yo'llar orqali bog'langanligini *bilishmaydi*.

Biz Opuztazerdagi yo'l tarmog'ining **zichligi** deb δ ning shunday qiymatiga aytamizki, unda har 3 ta har xil diqqatga sazovor joyning orasida eng kamida δ ta yo'l mavjud bo'ladi. Boshqacha qilib aytganda, har bir (u, v, w) $0 \leq u < v < w < N$ uchlik diqqatga sazovor joylar uchun $(u, v), (v, w)$ va (u, w) diqqatga sazovor juftliklar orasida eng kamida δ ta juftlik yo'l orqali bog'langan bo'lsa.

Tashkilotchilar yo'l tarmog'ining zichligi kamida D bo'lishi uchun D musbat sonini *bilishadi*. E'tibor bering, D qiymati 3 dan oshmasligi kerak.

Tashkilotchilar Opuztazerdagi telefon dispetcheriga **qo'ng'iroq** qilib, ba'zi diqqatga sazovor joylar orasidagi yo'l aloqalari haqida ma'lumot olishlari mumkin. Har bir **qo'ng'iroq**da ikkita bo'sh bo'lmagan diqqatga sazovor joylar massivlari $[A[0], \ldots, A[P-1]]$ va $[B[0], \ldots, B[R-1]]$ shakllantirilishi kerak. Diqqatga sazovor joylar juftliklari har xil bo'lishi kerak, bunda:

- Har bir i va j ($0 \leq i < j < P$) uchun $A[i] \neq A[j]$;
- Har bir i va j ($0 \leq i < j < R$) uchun $B[i] \neq B[j]$;
- Har bir i ($0 \leq i < P$) va j ($0 \leq j < R$) uchun $A[i] \neq B[j]$.

Har bir qo'ng'iroq uchun dispetcher A massivdagi diqqatga sazovor joylarning biridan B massividagi diqqatga sazovor joylarning biriga yo'l bor yoki yo'qligini aytadi. Aniqrog'i, dispetcher A va B massivlarining har bir $i(0 \leq i < P)$ va $j(0 \leq j < R)$ juftligini ko'rib chiqib agar qaysidir $A[i]$ va $B[j]$ diqqatga sazovor joylari orasida yo'l mavjud bo'lsa `true` aks holda `false` qiymatlarini qaytaradi.

l uzunlikdagi **sayohat** deb shunday $t[0], t[1], \ldots, t[l-1]$ ketma-ketligiga aytiladiki, bunda barcha qiymatlar har xil va har bir $i(0 \leq i \leq l-2)$ uchun $t[i]$ va $t[i+1]$ - diqqatga sazovor joylar orasida yo'l mavjud bo'ladi.

l uzunlikdagi sayohat agarda $l+1$ uzunlikdagi sayohat topilmasa **eng uzun sayohat** deb ataladi.

Sizning vazifangiz dispetcherlarga qo'ng'iroq qilish orqali tashkilotchilarga Opuztazerdagi eng uzun sayohatni topishda yordam berishdan iborat.

Implement qilish uchun tafsilotlar

Siz quyidagi protsedurani implement qilishingiz kerak:

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

- N: Opuztazerdagi diqqatga sazovor joylar soni.
- D: Kafolatlangan yo'l tarmog'ining eng kichik zichligi.
- Bu protsedura eng uzun sayohatni ifodalovchi $t = [t[0], t[1], \ldots, t[l-1]]$ massivini qaytarishi kerak.
- Bu protsedura har bir test case uchun **bir necha marotaba** qo'ng'iroq qilishi mumkin.

Yuqorida aytilgan protsedura qo'ng'iroqlarni quyidagi funksiyaga qilishi mumkin:

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

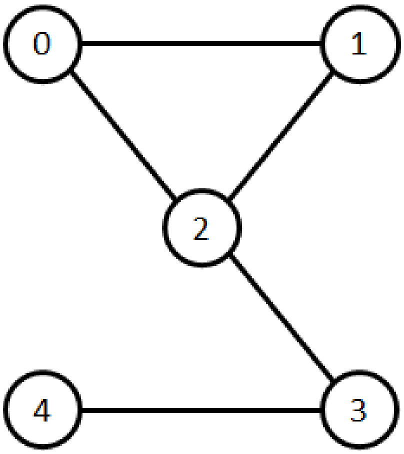
- A: har xil diqqatga sazovor joylardan tashkil topgan bo'sh bo'lmagan massiv.
- B: har xil diqqatga sazovor joylardan tashkil topgan bo'sh bo'lmagan massiv.
- A va B har xil elementlardan tashkil topgan bo'lishi kerak.
- Bu funksiya A massividagi qaysidir diqqatga sazovor joy B massividagi qaysidir bir diqqatga sazovor joy orasida yo'l mavjud bo'lsa `true` aks holda `false` qiymatini qaytaradi.
- Bu funksiyadan bitta eng uzun sayohatni aniqlashda ko'pi bilan $32\,640$ marotaba, umumiy hisobda ko'pi bilan $150\,000$ marotaba foydalanish mumkin.
- Bu funksiyaga murojaat qilish davomida A va B massivlarining umumiy uzunligi $1\,500\,000$ dan oshmasligi kerak.

Grayder **moslashuvchan emas**. Har bir jo'natilgan yechim bir xil test case lar bilan tekshiriladi.

Examples

Example 1

Consider a scenario in which $N = 5$, $D = 1$, and the road connections are as shown in the following figure:



The procedure `longest_trip` is called in the following way:

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

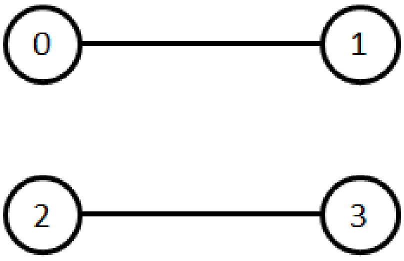
The procedure may make calls to `are_connected` as follows.

Call	Pairs connected by a road	Return value
`are_connected([0], [1, 2, 4, 3])`	$(0, 1)$ and $(0, 2)$	`true`
`are_connected([2], [0])`	$(2, 0)$	`true`
`are_connected([2], [3])`	$(2, 3)$	`true`
`are_connected([1, 0], [4, 3])`	none	`false`

After the fourth call, it turns out that *none* of the pairs $(1, 4)$, $(0, 4)$, $(1, 3)$ and $(0, 3)$ is connected by a road. As the density of the network is at least $D = 1$, we see that from the triplet $(0, 3, 4)$, the pair $(3, 4)$ must be connected by a road. Similarly to this, landmarks 0 and 1 must be connected.

At this point, it can be concluded that $t = [1, 0, 2, 3, 4]$ is a trip of length 5, and that there does not exist a trip of length greater than 5. Therefore, the procedure `longest_trip` may return $[1, 0, 2, 3, 4]$.

Consider another scenario in which $N = 4$, $D = 1$, and the roads between the landmarks are as shown in the following figure:



The procedure `longest_trip` is called in the following way:

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

In this scenario, the length of a longest trip is 2. Therefore, after a few calls to procedure `are_connected`, the procedure `longest_trip` may return one of $[0, 1]$, $[1, 0]$, $[2, 3]$ or $[3, 2]$.

Example 2

Subtask 0 contains an additional example test case with $N = 256$ landmarks. This test case is included in the attachment package that you can download from the contest system.

Constraints

- $3 \leq N \leq 256$
- The sum of N over all calls to `longest_trip` does not exceed $1\,024$ in each test case.
- $1 \leq D \leq 3$

Subtasks

1. (5 points) $D = 3$
2. (10 points) $D = 2$
3. (25 points) $D = 1$. Let $l^\star$ denote the length of a longest trip. Procedure `longest_trip` does not have to return a trip of length $l^\star$. Instead, it should return a trip of length at least $\left\lceil \frac{l^\star}{2} \right\rceil$.
4. (60 points) $D = 1$

In subtask 4 your score is determined based on the number of calls to procedure `are_connected` over a single invocation of `longest_trip`. Let q be the maximum number of calls among all invocations of `longest_trip` over every test case of the subtask. Your score for this subtask is calculated according to the following table:

Condition	Points
$2\,750 < q \leq 32\,640$	20
$550 < q \leq 2\,750$	30
$400 < q \leq 550$	45
$q \leq 400$	60

If, in any of the test cases, the calls to the procedure `are_connected` do not conform to the constraints described in Implementation Details, or the array returned by `longest_trip` is incorrect, the score of your solution for that subtask will be 0.

Sample Grader

Let C denote the number of scenarios, that is, the number of calls to `longest_trip`. The sample grader reads the input in the following format:

- line 1: C

The descriptions of C scenarios follow.

The sample grader reads the description of each scenario in the following format:

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

Here, each U_i ($1 \leq i < N$) is an array of size i, describing which pairs of landmarks are connected by a road. For each i and j such that $1 \leq i < N$ and $0 \leq j < i$:

- if landmarks j and i are connected by a road, then the value of $U_i[j]$ should be 1;
- if there is no road connecting landmarks j and i, then the value of $U_i[j]$ should be 0.

In each scenario, before calling `longest_trip`, the sample grader checks whether the density of the road network is at least D. If this condition is not met, it prints the message `Insufficient Density` and terminates.

If the sample grader detects a protocol violation, the output of the sample grader is `Protocol Violation: <MSG>`, where `<MSG>` is one of the following error messages:

- `invalid array`: in a call to `are_connected`, at least one of arrays A and B
  - is empty, or
  - contains an element that is not an integer between 0 and $N-1$, inclusive, or
  - contains the same element at least twice.
- `non-disjoint arrays`: in a call to `are_connected`, arrays A and B are not disjoint.
- `too many calls`: the number of calls made to `are_connected` exceeds $32\,640$ over the current invocation of `longest trip`, or exceeds $150\,000$ in total.
- `too many elements`: the total number of landmarks passed to `are_connected` over all calls exceeds $1\,500\,000$.

Otherwise, let the elements of the array returned by `longest_trip` in a scenario be $t[0], t[1], \ldots, t[l-1]$ for some nonnegative l. The sample grader prints three lines for this scenario in the following format:

- line 1: l
- line 2: $t[0] \ t[1] \ \ldots \ t[l-1]$
- line 3: the number of calls to `are_connected` over this scenario

Finally, the sample grader prints:

- line $1 + 3 \cdot C$: the maximum number of calls to `are_connected` over all calls to `longest_trip`