Сфинксийн оньсого

Агуу Сфинкс нь нэгэн оньсого юм. Танд N оройтой граф өгөгдсөн. Оройнуудыг нь 0 -ээс $N-1$ хүртэл дугаарласан. Энэхүү граф нь 0-ээс $M-1$ хүртэл дугаарласан M ирмэгтэй байна. Ирмэг бүр нь тодорхой хос оройг холбох бөгөөд хоёр чиглэлтэй байдаг. Тодруулбал, 0 -ээс $M-1$ (хил дээрх утгуудыг оруулаад) хүртэлх j бүрийн хувьд $X[j]$ ба $Y[j]$ оройнууд холбогдсон байна. Аливаа хос оройг холбосон ирмэг дээд тал нь нэг л байдаг. Хэрэв хоёр орой ирмэгээр холбогдсон тэдгээрийг **хөрш** гэж нэрлэдэг.

Дараалсан хоёр орой v_l ба v_{l+1} ($0 \leq l < k$ байх l бүрийн хувьд) бүр нь хөрш байдаг $v_0, v_1, \ldots, v_k$ ($k \geq 0$) оройнуудын дараалыг **зам** гэж нэрлэдэг. Энэхүү $v_0, v_1, \ldots, v_k$ зам нь v_0 ба v_k хоёр оройг **холбодог** гэж бид ярина. Танд өгөгдсөн граф дээр хос орой бүр ямар нэгэн замаар холбогдсон байна.

0 -ээс N хүртэл дугаарлагдсан $N+1$ өнгө байна. Өнгө N нь онцгой бөгөөд үүнийг **Сфинксийн өнгө** гэж нэрлэдэг. Орой болгонд өнгө өгөгдсөн. Тодруулбал, i-р орой ($0 \leq i < N$) нь $C[i]$ өнгөтэй байна. Олон орой нь ижил өнгөтэй байж болно. Мөн ямар ч оройд хуваарилагдаагүй өнгө байж болно. $0 \leq C[i] < N$ ($0 \leq i < N$) байх ямар ч орой сфинксийн өнгөтэй байдаггүй.

Хэрэв бүх орой нь ижил өнгөтэй, өөрөөр хэлбэл $C[v_l] = C[v_{l+1}]$ ($0 \leq l < k$ байх l бүрийн хувьд) байдаг $v_0, v_1, \ldots, v_k$ ($k \geq 0$) замыг **монохроматик** гэе. Нэмж дурдахад бид зөвхөн монохроматик замаар холбогдсон p ба q ($0 \leq p < N$, $0 \leq q < N$) оройнуудыг нэгэн ижил **монохроматик компонент** дотор байна гэж үзнэ.

Та орой болон ирмэгүүдийг мэддэг, гэхдээ орой бүр ямар өнгөтэй болохыг мэдэхгүй байна. Та **дахин будах туршилт** хийх замаар оройнуудын өнгийг мэдэхийг хүсч байна.

Дахин будах туршилтанд, та олон оройн өнгийг дур мэдэн өөрчилж болно. Тодруулбал, дахин будах туршилт хийхдээ та эхлээд $0 \leq i < N$ байх i бүрийн хувьд $E[i]$ нь -1 болон N хооронд (хилийн утгууд оруулан) байх N хэмжээтэй E массивыг сонгоно . Ингэснээр i орой бүрийн өнгө нь $S[i]$ болох ба $S[i]$ утга нь:

- $C[i]$, хэрэв $E[i] = -1$ бол i -ийн анхны өнгө, эсвэл
- $E[i]$, үгүй бол.

Энэ нь та дахин будахдаа Сфинксийн өнгийг ашиглаж болно гэсэн үг гэдгийг анхаарна уу.

Эцэст нь, Агуу Сфинкс i орой бүрийн өнгийг $S[i]$ ($0 \leq i < N$) болгон тохируулсны дараа граф дахь **монохроматик компонент**-ын тоог зарладаг. Шинэ өнгийг зөвхөн энэ дахин будах туршилтад ашигласан тул **туршилт дуусcаны дараа бүх оройн өнгө анхных руугаа буцна**.

Таны даалгавар бол хамгийн ихдээ $2\,750$ өнгө өөрчлөх туршилт хийж граф дахь оройнуудын өнгийг тодорхойлох явдал юм. Хэрэв та хөрш орой бүрийг ижил өнгөтэй эсэхийг зөв тодорхойлж чадвал хэсэгчилсэн оноо авч болно.

Хэрэгжүүлэлтийн мэдээлэл

Та дараах функцийг хэрэгжүүлэх ёстой:

```
std::vector<int> find_colours(int N,
    std::vector<int> X, std::vector<int> Y)
```

- N: графын оройн тоо.
- X, Y: ирмэгүүдийг тодорхойлох M урттай массивууд.
- Энэхүү функц нь графын оройнуудын өнгийг илэрхийлэх N урттай G массив буцаана.
- Энэ функц нь тест бүр дээр яг нэг удаа дуудагдана.

Дээрх функц нь дахин будалт хийх туршилтыг гүйцэтгэх дараах функцийг дуудна.

```
int perform_experiment(std::vector<int> E)
```

- E : оройнуудын өнгийг хэрхэн өөрчлөхийг заасан N урттай массив.
- Энэ функц нь E -ын утгаар оройнуудыг дахин будсаны дараа монохроматик компонентуудын тоог буцаана.
- Энэ функцийг хамгийн ихдээ $2\,750$ удаа дуудаж болно.

Grader нь **дасан зохицох чадваргүй**, өөрөөр хэлбэл `find_colours` функцийг дуудахаас өмнө оройнуудын өнгийг засдаг.

Хязгаарлалтууд

- $2 \leq N \leq 250$
- $N - 1 \leq M \leq \frac{N \cdot (N-1)}{2}$
- $0 \leq j < M$ байх j бүрийн хувьд $0 \leq X[j] < Y[j] < N$ байна.
- $0 \leq j < k < M$ байх j ба k бүрийн хувьд $X[j] \neq X[k]$ эсвэл $Y[j] \neq Y[k]$ байна.
- Хос орой бүр ямар нэг замаар холбогдсон байна.
- $0 \leq i < N$ байх i бүрийн хувьд $0 \leq C[i] < N$ байна.

Дэд бодлогууд

Дэд бодлого	Оноо	Нэмэлт хязгаарлалтууд
1	3	$N = 2$
2	7	$N \leq 50$
3	33	Граф нь нэг зам: $M = N - 1$ бөгөөд j ба $j + 1$ $(0 \leq j < M)$ оройнууд нь хөрш байна.
4	21	Граф нь гүйцэд: $M = \frac{N \cdot (N-1)}{2}$ бөгөөд аливаа хоёр орой нь хөрш байна.
5	36	Нэмэлт хязгаарлалт байхгүй.

Хэрэв таны программ хөрш орой бүрийн хос ижил өнгөтэй эсэхийг зөв тодорхойлсон бол дэд бодлого бүрт хэсэгчилсэн оноо авах боломжтой.

Илүү тодруулан хэлэхэд, хэрэв бүх тестийн тохиолдлуудад `find_colours` -ын буцаасан G массив нь C массивтай яг адилхан байвал та дэд бодлогын оноог бүхэлд нь авна (өөрөөр хэлбэл $0 \leq i < N$ байх бүх i -ийн хувьд $G[i] = C[i]$). Үгүй бол, дараах нөхцөлүүдийг бүх тестийн тохиолдлуудад хангагдсан тохиолдолд дэд бодлогын онооны 50% авна:

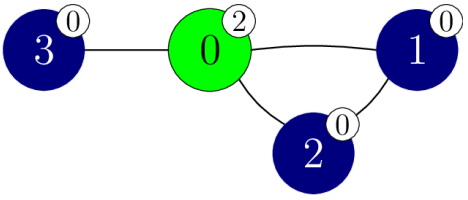
- $0 \leq i < N$ байх i бүрийн хувьд $0 \leq G[i] < N$ бол;
- $0 \leq j < M$ байх j бүрийн хувьд $G[X[j]] = G[Y[j]]$ ба $C[X[j]] = C[Y[j]]$ бол.

Жишээ

Дараах дуудалтыг хийсэн гэж үзье.

```
find_colours(4, [0, 1, 0, 0], [1, 2, 2, 3])
```

Энэ жишээний хувьд оройнуудын (далд) өнгийг $C = [2, 0, 0, 0]$ утгатай өгсөн гэж бодъё. Энэ хувилбарыг дараах зурагт үзүүлэв. Өнгө нь орой тус бүр дээр хавсаргасан цагаан дугуй шошгон дээр тоогоор илэрхийлэгдсэн байна.



Энэ функц нь дараах `perform_experiment` функцийг дуудаж болно.

```
perform_experiment([-1, -1, -1, -1])
```

Энэ дуудлагад бүх оройнууд анхны өнгөө хадгалдаг тул ямар ч оройг дахин будахгүй.

1 ба 2 оройг авч үзье. Тэд хоёулаа 0 өнгөтэй бөгөөд $1, 2$ зам нь монохроматик зам юм. Үүний үр дүнд 1 ба 2 оройнууд ижил монохроматик компонент дотор байна.

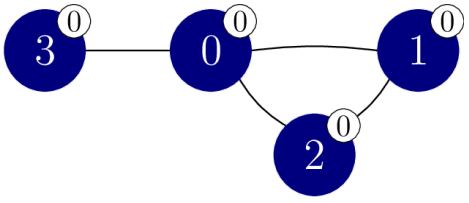
1 ба 3 оройг авч үзье. Хэдийгээр хоёулаа 0 өнгөтэй боловч тэдгээрийг холбосон монохроматик зам байхгүй тул тэд өөр өөр монохроматик компонентод хамаарч байна.

Ерөнхийдөө 3 монохроматик компонент байх ба оройнууд нь $\{0\}$, $\{1, 2\}$, $\{3\}$. Тиймээс энэ дуудалт 3 утга буцаана.

Одоо энэ функц нь дараах байдлаар `perform_experiment` функцийг дуудаж болно.

```
perform_experiment([0, -1, -1, -1])
```

Энэ дуудлагад зөвхөн 0 оройг 0 өнгөтэй болгож өөрчилсөн бөгөөд үүний үр дүнд дараах зурагт үзүүлсэн өнгө гарч ирнэ.

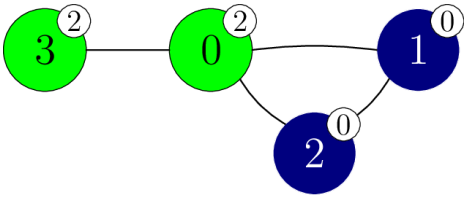


Бүх оройнууд ижил монохроматик компонентод хамаарах тул энэ дуудалт 1 утга буцаана. Одоо бид 1 , 2 , 3 оройнууд 0 өнгөтэй байна гэж дүгнэж болно.

Энэ функц нь `perform_experiment` функцийг дараах байдлаар дуудаж болно.

```
perform_experiment([-1, -1, -1, 2])
```

Энэ дуудлагад 3 оройг 2 өнгөтэй болгож өөрчилсөн бөгөөд үүний үр дүнд дараах зурагт үзүүлсэн өнгө гарч ирнэ.



Энэ дуудлага нь 2 буцаана, учир нь 2 монохроматик компонент байдаг ба оройнууд нь $\{0, 3\}$ болон $\{1, 2\}$. 0 орой нь 2 өнгөтэй байна гэж бид дүгнэж болно.

Дараа нь `find_colours` функц $[2,0,0,0]$ массивыг буцаана. $C=[2,0,0,0]$ тул бүтэн оноо өгнө.

Онооны 50% ийг өгөх хэд хэдэн буцаах утгууд байдгийг анхаарна уу, жишээ нь $[1,2,2,2]$ эсвэл $[1,2,2,3]$.

Жишээ Grader

Оролтын формат:

```
N  M
C[0]  C[1] ... C[N-1]
X[0]  Y[0]
X[1]  Y[1]
...
X[M-1]  Y[M-1]
```

Гаралтын формат:

```
L  Q
G[0]  G[1] ... G[L-1]
```

Энд L нь `find_colours` -ын буцаасан G массивын урт бөгөөд Q нь `perform_experiment` дуудлагын тоо юм.