Favvora Bog'lari

Istirohat bog'ida n ta **favvoralar** bo'lib, ular 0 dan $n-1$ gacha raqamlangan. Favvoralarni ikki o'lchovli fazodagi nuqtalar kabi ifodalaymiz. Ya'ni, i($0 \leq i \leq n-1$)-favvora $(x[i], y[i])$ nuqtani bildiradi va $x[i]$ va $y[i]$ **juft sonlardir**. Favvoralar turli xil nuqtalarda joylashgan.

Arxitektor Timotiga bir nechta **yo'llar** va har bir yo'l uchun **o'rindiqlar** qurish vazifasi topshirildi. Bunda yo'l deb uzunligi 2 bo'lgan **gorizontal** yoki **vertikal** kesmaga aytiladi. Bunda yo'l har doim ikita har xil fontanni bog'lab turadi. Yo'llar shunday qurilishi kerakki, har qanday fontandan ixtiyoriy fontanga yo'llar orqali borish imkoni bo'lsin. Boshlang'ich vaqtda bog'da hech qanday yo'llar yo'q.

Har bir yo'l uchun, **aynan** bitta o'rindiq bog'ga qo'yilishi kerak va u o'sha yo'lga **biriktirilgan** bo'lishi kerak. Har o'rindiq shunday (a, b) nuqtada joylashtirilishi kerakki, a va b **toq butun sonlar** bo'lishi kerak. Hamma o'rindiqlar lokatsiyasi **har xil** bo'lishi kerak. (a, b) nuqtadagi o'rindiq qaysidir yo'lga biriktirilishi mumkin, qachonki yo'lning **ikkala** tugash nuqtasi ham $(a-1, b-1)$, $(a-1, b+1)$, $(a+1, b-1)$ va $(a+1, b+1)$ lar orasida bo'lsa.

Masalan, $(3, 3)$ nuqtadagi o'rindiq faqatgina $(2, 2)$ – $(2, 4)$, $(2, 4)$ – $(4, 4)$, $(4, 4)$ – $(4, 2)$, $(4, 2)$ – $(2, 2)$ kabi kesmali yo'llarga biriktirilishi mumkin.

Timotyga har bir fontandan qolgan har biriga borsa bo'ladigan qilib yo'llar qurish va ularga o'rindiqlar biriktirish mumkinligini aniqlashga yordam bering. Agar iloji bo'lsa unga birorta yechim taklif qiling, agar bunday yechimlar bir nechta bo'lsa har qanday yechim taklif qilish mumkin

Implementatsiya tafsilotlari

Quyidagi funksiyani implementatsiya qilish kerak:

```
int construct_roads(int[] x, int[] y)
```

- x, y: uzunligi n bo'lga ikkita massiv. Har bir i ($0 \leq i \leq n-1$) uchun, i-favvora $(x[i], y[i])$-nuqtada, bu yerda $x[i]$ va $y[i]$ ikkita har xil son.
- Agar qurish iloji bo'lsa bu funksiya aniq bir marta `build` funksiyasini chaqirishi kerak(pastroqqa qarang) yechim haqida aytish uchun, va funksiyaning o'zi 1 qaytarishi kerak.
- Aks holda, funksiya 0 qaytarishi kerak `build` ni biror marta ham chaqirmasdan.
- Bu funksiya aynan bir marta chaqiriladi.

Sizning implementatsiyangiz yo'llar va o'rindiqlarni qurilishi yechimi bilan ta'minlash uchun quyidagi prosedurani chaqirishi mumkin.

```
void build(int[] u, int[] v, int[] a, int[] b)
```

- u, v: uzunligi m bo'lgan ikkita massiv, yo'llar qurilishini ko'rsatadi. Bu yo'llar 0 dan $m-1$ gacha raqamlanadi. Har bir j ($0 \leq j \leq m-1$) uchun, $j-$yo'l $u[j]$ va $v[j]$-favvoralarni bog'laydi. Har bir yo'l uzunligi 2 bo'lgan vertikal yoki gorizontal kesma bo'lishi kerak. Har qanday ikkita yo'l faqatgina bitta umumiy nuqtaga ega bo'lishi mumkin va u ham faqatgina favvora bo'la oladi. Yo'llar qurilganidan keyin har bir favvoradan boshqa har biriga yo'llar orqali borish imkoni bo'lsin.
- a, b: uzunligi m bo'lgan ikkita massiv, o'rindiqlarni ifodalaydi. Har bir j ($0 \leq j \leq m-1$) uchun, $(a[j], b[j])$ nuqtada joylashgan, va j-yo'lga biriktirilgan. Hech qaysi ikki o'rindiq bir xil joylashuvga ega emas. Hamma yo'llarning o'rindiqlari har xil.

Misollar

Misol 1

Quyidagi funksiyani implementatsiya qilish kerak:

```
construct_roads([4, 4, 6, 4, 2], [4, 6, 4, 2, 4])
```

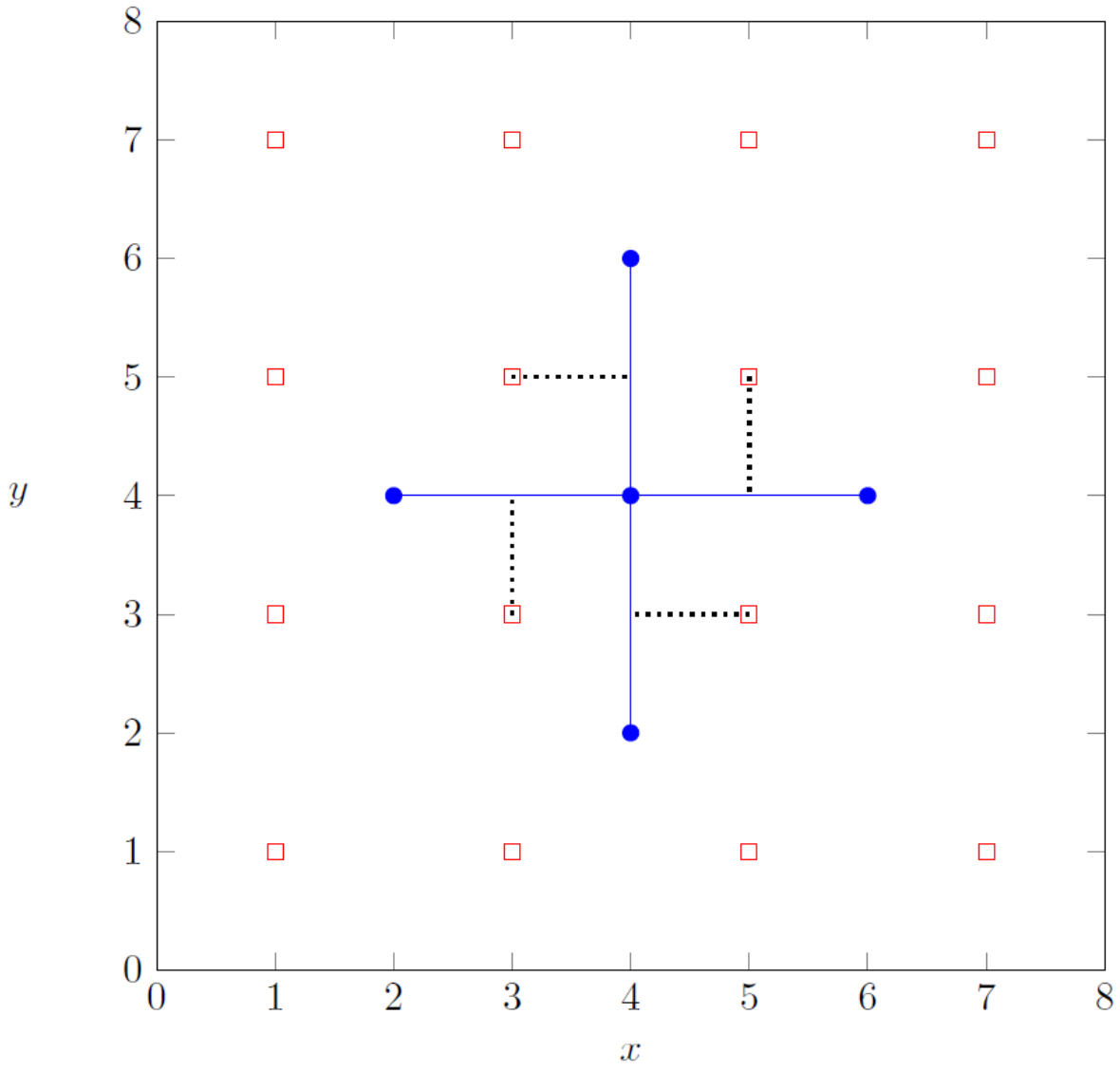
Bu anglatadiki 5 ta favvora bor:

- 0-favvora $(4, 4)$-da joylashgan,
- 1-favvora $(4, 6)$-da joylashgan,
- 2-favvora $(6, 4)$-da joylashgan,
- 3-favvora $(4, 2)$-da joylashgan,
- 4-favvora $(2, 4)$-da joylashgan.

Quyidagi 4 ta yo'lni qurish mumkin, bu yerda yo'llar favvoralarni bog'laydi va o'rindiqlar biriktirliadi.

Yo'l indeksi	Yo'l bog'laydigan favvoralar indeksi	Biriktirilgan o'rindiqlar joylashuvi
0	$0, 2$	$(5, 5)$
1	$0, 1$	$(3, 5)$
2	$3, 0$	$(5, 3)$
3	$4, 0$	$(3, 3)$

Bu yechim quyidagi diagrammaga mos keladi:



Yechim bilan ta'minlash uchun, `construct_roads` quyidagi chaqiruvni amalga oshirishi kerak:

- `build([0, 0, 3, 4], [2, 1, 0, 0], [5, 3, 5, 3], [5, 5, 3, 3])`

Funksiya 1 qaytarishi kerak.

Bu holatda, bu yerda bir nechta yechimlar bor, ularning ixtiyoriysi to'g'ri deb hisoblanadi. Masalan, quyidagi chaqiruv ham to'g'ri hisoblanadi `build([1, 2, 3, 4], [0, 0, 0, 0], [5, 5, 3, 3], [5, 3, 3, 5])` va funksiya 1 qaytarishi kerak.

Example 2

Quyidagi chaqiruvni ko'raylik:

```
construct_roads([2, 4], [2, 6])
```

0-favvora $(2, 2)$da joylashgan va 1-favvora $(4, 6)$da joylashgan. Bu yerda hamma talabga javob beradigan yo'llar qurish imkoni yo'qligi sababli `construct_roads` funksiyasi `build` ni chaqirmasdan

0 qaytarishi kerak.

Chegaralar

- $1 \leq n \leq 200\,000$
- $2 \leq x[i], y[i] \leq 200\,000$ (har bir $0 \leq i \leq n-1$ uchun)
- $x[i]$ va $y[i]$lar juft sonlar (har bir $0 \leq i \leq n-1$ uchun).
- Hech qaysi ikkita favvoralar bitta nuqtada emas.

Qism masalalar

1. (5 points) $x[i] = 2$ (har bir $0 \leq i \leq n-1$ uchun)
2. (10 points) $2 \leq x[i] \leq 4$ (har bir $0 \leq i \leq n-1$ uchun)
3. (15 points) $2 \leq x[i] \leq 6$ (har bir $0 \leq i \leq n-1$ uchun)
4. (20 points) har bir favvoradan boshqasiga bora oladigan qilib yo'llarni qurishni ko'pi bilan bitta yo'li bor.
5. (20 points) 2×2 kvadrat hosil qiladigan to'rtta favvoralar mavjud emas.
6. (30 points) Qo'shimcha chegaralar yo'q

Namunaviy Grader

Grader kiruvchi ma'lumotlarni quyidagi formatda o'qiydi:

- 1-qator: n
- $2 + i$-qatorlar ($0 \leq i \leq n-1$): $x[i]$ $y[i]$

Grader chiqaradigan ma'lumotlar quyidagi formatda bo'ladi:

- 1-qator: `construct_roads` qaytargan qiymat

`construct_roads` qaytargan qiymat 1 bo'lsa va `build(u, v, a, b)` chaqirilgan bo'lsa, grader qo'shimchasiga quyidagilarni chiqaradi:

- 2-qator: m
- $3 + i$-qatorlar ($0 \leq i \leq m-1$): $u[i]$ $v[i]$ $a[i]$ $b[i]$