Хурдны замын төлбөр

Японд хотууд нь сүлжээ бүхий хурдны замуудаар холбогдсон байдаг. Энэхүү сүлжээ нь N хот болон M хурдны замаас бүрддэг. Хурдны зам бүр ялгаатай хоёр хотыг холбодог. Аль ч хоёр хурдны зам ижил хос хотуудыг холбодоггүй. Хотууд нь 0-ээс $N-1$ хүртэл, хурдны замууд 0-ээс $M-1$ хүртэл дугаарлагдана. Та хурдны замаар аль ч чиглэлд явж болох бөгөөд та энэхүү хурдны замуудаар дамжин аль ч хотоос аль ч хот руу хүрч чадна.

Аливаа хурдны замаар явахад зам ашиглалтын төлбөр (цаашид замын төлбөр гэнэ) авдаг. Тухайн замын төлбөр нь **замын ачааллаас** хамаардаг. Замын ачаалал нь **хөнгөн** эсвэл **хүнд** байна. Хэрэв замын ачаалал хөнгөн байвал замын төлбөр A иен, хүнд байвал B иен байдаг. $A < B$ байна. Тэмдэглэж хэлэхэд та A болон B-ийн утгуудыг мэдэж байгаа.

Танд бүх хурдны замын ачааллын мэдээллийг өгөхөд дээрх нөхцөлийн дагуу S хотоос эхлэн T хотод очих хамгийн бага зардлыг олдог төхөөрөмж байгаа.

Гэхдээ энэхүү төхөөрөмж нь туршилтын шатандаа байгаа учраас S болон T-ийн утгууд нь тогтмол (өөрчлөгдөх боломжгүйгээр тохируулсан) бөгөөд танд мэдэгдэхгүй байгаа. Таны даалгавар бол S болон T-г олох явдал юм. Эдгээрийг олохын тулд та хэд хэдэн замын хөдөлгөөний нөхцөл байдлыг тухайн төхөөрөмжөөр турших юм. Эдгээр туршилтыг хийх нь зардал ихтэй байдаг учраас та энэхүү машиныг ихээр ашиглахыг хүсэхгүй байгаа.

Хэрэгжүүлэлтийн мэдээлэл

Та дараах функцийг хэрэгжүүлнэ.

```
find_pair(int N, int[] U, int[] V, int A, int B)
```

- `N`: хотуудын тоо.
- `U` ба `V`: M урттай тоон дарааллууд байх бөгөөд i ($0 \leq i \leq M-1$) хувьд i дүгээр хурдны зам нь `U[i]` болон `V[i]` хотуудыг холбосныг илэрхийлнэ.
- `A`: Хөнгөн ачаалалтай хурдны замаас авах төлбөр.
- `B`: Хүнд ачаалалтай хурдны замаас авах төлбөр.
- Энэхүү функцийг тест бүрд яг нэг удаа дуудна.
- Тэмдэглэж хэлэхэд, M нь тоон дарааллуудын урт бөгөөд түүнийг хэрхэн олох талаарх мэдээллийг хэрэгжүүлэлтийн тэмдэглэл (Implementation Notice)-д

тодотгож өгсөн.

Энэхүү `find_pair` функц нь дараах функцүүдийг дууддаг байна:

```
int64 ask(int[] w)
```

- `w`-ийн урт M байх ёстой. `w` дараалал замын хөдөлгөөний мэдээллийг илэрхийлэх ёстой.
- i $(0 \leq i \leq M-1)$ бүрийн хувьд `w[i]` нь i дүгээр хурдны замын ачааллыг илэрхийлэх ёстой. `w[i]`-ийн утга 0 эсвэл 1 байна.
  - `w[i] = 0` бол i дүгээр хурдны замын ачаалал хөнгөн байна.
  - `w[i] = 1` бол i дүгээр хурдны замын ачаалал хүнд байна.
- Энэхүү функц замын хөдөлгөөний ачаалал `w` байх үед S-ээс T-д хүрэхэд байж болох нийт замын төлбөрийн хамгийн багыг буцаана.
- Энэхүү функц хамгийн ихдээ 100 удаа дуудагдана (тест бүрийн хувьд).

`find_pair` функц хариугаа дараах функцээр мэдүүлнэ:

```
answer(int s, int t)
```

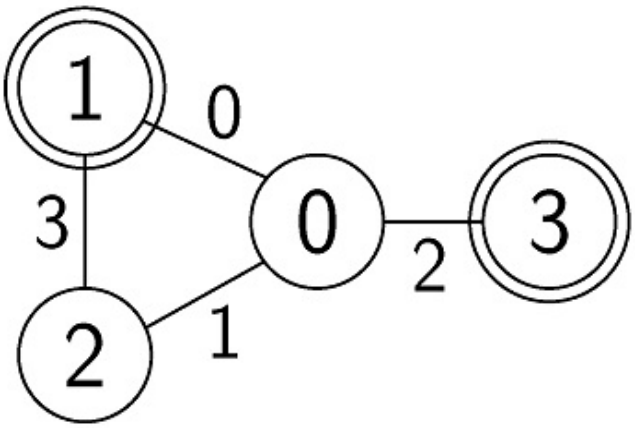
- `s` болон `t` нь S болон T хос байна (байрлал нь хамаагүй).
- Энэхүү функцийг яг нэг удаа дуудна.

Хэрэв дээрх дурдсан нөхцөлүүдийн аль нэгийг зөрчвөл та **Wrong answer** хариултыг авах юм. Үгүй бол, **Accepted** хариулт авах ба `ask` функцийн дуудалтаас хамааруулан оноог тань бодно (Дэд бодлогыг үзнэ үү).

Жишээ

$N = 4$, $M = 4$, $U = [0, 0, 0, 1]$, $V = [1, 2, 3, 2]$, $A = 1$, $B = 3$, $S = 1$, ба $T = 3$ байг.

Шалгагч `find_pair(4, [0, 0, 0, 1], [1, 2, 3, 2], 1, 3)` гэж дуудна.



Дээрх зурагт i дүгээр ирмэг i дүгээр хурдны замыг илэрхийлнэ. `ask` функцийн зарим боломжит дуудалт болон харгалзах утга буцаалтуудыг доорх хүснэгтэд үзүүллээ:

Дуудалт	Буцаалт
`ask([0, 0, 0, 0])`	2
`ask([0, 1, 1, 0])`	4
`ask([1, 0, 1, 0])`	5
`ask([1, 1, 1, 1])`	6

`ask([0, 0, 0, 0])` дуудалтад бүх хурдны замын ачаалал нь хөнгөн учир бүх замын төлбөр 1 байна. Хамгийн хямд $S = 1$-ээс $T = 3$-д хүрэх зам нь $1 \to 0 \to 3$. Энэ замын нийт замын төлбөр нь 2 юм. Иймд, энэ функц 2 утгыг буцаана.

Зөв хариултын хувьд `find_pair` функц `answer(1, 3)` эсвэл `answer(3, 1)` гэж дуудах ёстой.

Энэхүү жишээ нь zip файлд `sample-01-in.txt` нэрээр хадгалагдсан. Бусад жишээнүүд zip файлд бий.

Хязгаарлалтууд

- $2 \le N \le 90\,000$
- $1 \le M \le 130\,000$
- $1 \le A < B \le 1\,000\,000\,000$
- $0 \le i \le M - 1$ бүрийн хувьд
  - $0 \le U[i] \le N - 1$
  - $0 \le V[i] \le N - 1$
  - $U[i] \ne V[i]$
- $(U[i], V[i]) \ne (U[j], V[j])$ ба $(U[i], V[i]) \ne (V[j], U[j])$ $(0 \le i < j \le M - 1)$
- Аль ч хотоос аль ч хот уруу хурдны замуудыг ашиглан хүрч болдог.
- $0 \le S \le N - 1$
- $0 \le T \le N - 1$
- $S \ne T$

Энэхүү бодлогын шалгагч нь дасан зохицдоггүй буюу таны асуулгуудаас хамааран S болон T-ийн утгууд өөрчлөгддөггүй гэсэг үг юм.

Дэд бодлогууд

1. (5 оноо) S эсвэл T нь 0, $N \le 100$, $M = N - 1$
2. (7 оноо) S эсвэл T нь 0, $M = N - 1$

3. (6 оноо) $M = N - 1$, $U[i] = i$, $V[i] = i + 1$ $(0 \le i \le M - 1)$
4. (33 оноо) $M = N - 1$
5. (18 оноо) $A = 1$, $B = 2$
6. (31 оноо) Нэмэлт хязгаарлалт байхгүй.

Тест бүрийн хувьд **Accepted** хариу авсан бөгөөд X удаа `ask` функцийг дуудсан гэе. Тэгвэл таны оноо P дэд бодлогоос хамааран дараах байдлаар бодогдоно:

- Дэд бодлого 1. $P = 5$.
- Дэд бодлого 2. Хэрэв $X \le 60$, $P = 7$. Үгүй бол $P = 0$.
- Дэд бодлого 3. Хэрэв $X \le 60$, $P = 6$. Үгүй бол $P = 0$.
- Дэд бодлого 4. Хэрэв $X \le 60$, $P = 33$. Үгүй бол $P = 0$.
- Дэд бодлого 5. Хэрэв $X \le 52$, $P = 18$. Үгүй бол $P = 0$.
- Дэд бодлого 6.
  - Хэрэв $X \le 50$, $P = 31$.
  - Хэрэв $51 \le X \le 52$, $P = 21$.
  - Хэрэв $53 \le X$, $P = 0$.

Тэмдэглэж хэлэхэд таны дэд бодлогуудын оноо нь тухайн дэд бодлогын тестүүдийн хамгийн бага оноогоор тооцогдоно.

Жишээ шалгагч

Жишээ шалгагч нь дараах байдлаар өгөгдлийг уншина.

- Мөр 1: N M A B S T
- Мөр $2 + i$ $(0 \le i \le M - 1)$: $U[i]$ $V[i]$

Хэрэв **Accepted** хариуг авсан бол `ask` функцийн дуудалтын тоо `q`-г `Accepted: q` гэх байдлаар гаргана.

Хэрэв **Wrong Answer** хариуг авсан бол `Wrong Answer: MSG` гэж гаргана. `MSG`-ийн утга нь дараахын аль нэг байна:

- `answered not exactly once`: `answer` функцийг яг нэг удаа дуудаагүй байх.
- `w is invalid`: `ask` функцэд өгсөн `w`-ийн урт M биш эсвэл i $(0 \le i \le M - 1)$ бүрийн хувьд аль нэг `w[i]` нь 0 эсвэл 1 биш .
- `more than 100 calls to ask`: `ask` функцийг 100-аас их удаа дуудсан.
- `{s, t} is wrong`: `answer` функцэд буруу `s` болон `t`-г мэдүүлсэн бол.