Fountain Parks

Num parque próximo, existem n **fontes**, numeradas de 0 a $n-1$. Modelamos as fontes como pontos num plano bi-dimensional. Nomeadamente, a fonte i ($0 \leq i \leq n-1$) é um ponto $(x[i], y[i])$ onde $x[i]$ e $y[i]$ são **inteiros pares**. As localizações das fontes são todas distintas.

Timothy, o arquiteto, foi contratado para planear a construção de algumas **estradas** e para colocar um **banco** (de jardim) por cada estrada. Uma estrada é um segmento de reta **horizontal** ou **vertical** de comprimento 2, cujos pontos finais são duas fontes distintas. As estradas devem ser construídas de maneira a que seja possível viajar entre quaisquer duas fontes movendo-se através das estradas. No início não existe nenhuma estrada no parque.

Para cada estrada,**exatamente** um banco deve ser colocado no parque e deve ser **atribuído** a (isto é, fica voltado para) essa estrada. Cada banco deve ser colocar num ponto (a, b) tal que a e b sejam **inteiros ímpares**. As localizações dos bancos devem ser todas **distintas**. Um banco em (a, b) apenas pode ser atribuído a uma estrada se **ambos** os pontos finais da estrada estão entre $(a-1, b-1)$, $(a-1, b+1)$, $(a+1, b-1)$ e $(a+1, b+1)$. Por exemplo, um banco em $(3, 3)$ pode apenas ser atribuído a uma estrada, que deve ser um dos seguintes quatro segmentos de reta: $(2, 2)$ – $(2, 4)$, $(2, 4)$ – $(4, 4)$, $(4, 4)$ – $(4, 2)$, $(4, 2)$ – $(2, 2)$.

Ajuda o Timothy a determinar se é possível construir as estradas e colocar bancos satisfazendo todas as condições dadas em cima e, se tal for possível, indica uma solução válida. Se existirem múltiplas soluções válidas que satisfazem todas as condições, podes indicar qualquer uma delas.

Detalhes de Implementação

Deves implementar a seguinte função:

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

- x, y: dois arrays de tamanho n. Para cada i ($0 \leq i \leq n-1$), a fonte i é um ponto $(x[i], y[i])$, onde $x[i]$ e $y[i]$ são inteiros pares.
- Se a construção for possível, esta função deve fazer exactamente uma chamada a `build` (ver a seguir) para indicar uma solução, sendo que de seguida deve devolver 1.
- Caso contrário, a função deve devolver 0 sem fazer nenhuma chamada a `build`.
- Esta função é chamada exatamente uma vez.

A tua implementação pode chamar o seguinte procedimento para indicar uma construção de estradas e colocação de bancos válida:

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

- Seja m o número total de estradas a construir.
- u, v: dois arrays de tamanho m, representando as estradas a serem construídas. Estas estradas são numeradas de 0 a $m-1$. Para cada j ($0 \leq j \leq m-1$), a estrada j liga duas fontes $u[j]$ e $v[j]$. Cada estrada deve ser um segmento de reta horizontal ou vertical de comprimento 2. Duas estradas distintas podem ter no máximo um ponto em comum (uma fonte). Uma vez que as estradas estejam construídas, deve ser possível viajar entre duas quaisquer fontes usando as estradas.
- a, b: dois arrays de tamanho m, representando os bancos. Para cada j ($0 \leq j \leq m-1$), um banco é colocado em $(a[j], b[j])$, e é atribuído à estrada j. Dois bancos distintos não podem ficar na mesma localização.

Exemplos

Exemplo 1

Considera a seguinte chamada:

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

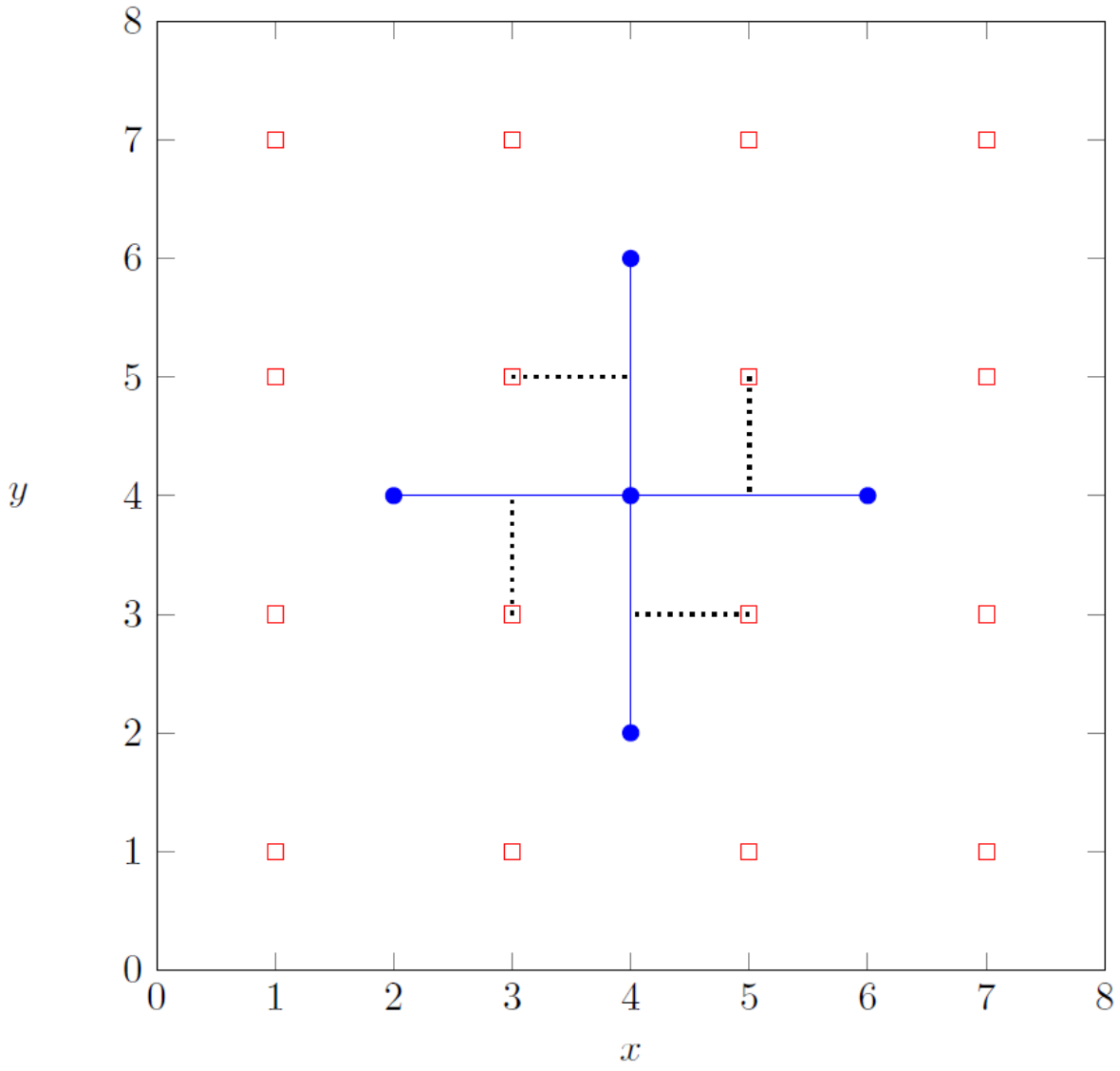
Isto significa que existem 5 fontes:

- fonte 0 localizada em $(4, 4)$,
- fonte 1 localizada em $(4, 6)$,
- fonte 2 localizada em $(6, 4)$,
- fonte 3 localizada em $(4, 2)$,
- fonte 4 localizada em $(2, 4)$.

É possível construír as seguintes 4 estradas, onde cada estrada liga duas fontes, e colocar os correspondentes bancos:

Estrada	Fontes que a estrada liga	Localização do banco atribuída a essa estrada
0	$0, 2$	$(5, 5)$
1	$0, 1$	$(3, 5)$
2	$3, 0$	$(5, 3)$
3	$4, 0$	$(3, 3)$

Esta solução corresponde ao seguinte diagrama:



Para reportar esta solução, `construct_roads` deve fazer a seguinte chamada:

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

Deve depois devolver 1.

Nota que neste caso existem múltiplas soluções que satisfazem todos os requisitos, e todas seriam consideradas corretas. Por exemplo, também será correto chamar `build([1, 2, 3, 4], [0, 0, 0, 0], [5, 5, 3, 3], [5, 3, 3, 5])` e depois devolver 1.

Exemplo 2

Considera a seguinte chamada:

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

A fonte 0 está localizada em $(2, 2)$ e a fonte 1 está localizada em $(4, 6)$. Como não existe maneira de construir as estradas de modo a satisfazer os requisitos, `construct_roads` deve

devolver 0 sem fazer nenhuma chamada a `build`.

Restrições

- $1 \leq n \leq 200\,000$
- $2 \leq x[i], y[i] \leq 200\,000$ (para $0 \leq i \leq n-1$)
- $x[i]$ e $y[i]$ são inteiros pares (para $0 \leq i \leq n-1$).
- Nunca existem duas fontes diferentes na mesma localização.

Subtarefas

1. (5 pontos) $x[i] = 2$ (para $0 \leq i \leq n-1$)
2. (10 pontos) $2 \leq x[i] \leq 4$ (para $0 \leq i \leq n-1$)
3. (15 pontos) $2 \leq x[i] \leq 6$ (para $0 \leq i \leq n-1$)
4. (20 pontos) Existe no máximo uma maneira de construir as estradas de tal modo que é possível viajar entre duas quaisquer fontes usando as estradas.
5. (20 pontos) Não existem quatro fontes que sejam os cantos de um quadrado de 2×2.
6. (30 pontos) Sem restrições adicionais.

Avaliador exemplo

O avaliador exemplo lê o input no seguinte formato:

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

O output do avaliador exemplo está no seguinte formato:

- linha 1: o valor devolvido por `construct_roads`

Se o valor devolvido por `construct_roads` for 1 e `build(u, v, a, b)` for chamado, o avaliador exemplo escreve também adicionalmente:

- linha 2: m
- linha $3+j$ ($0 \leq j \leq m-1$): $u[j]$ $v[j]$ $a[j]$ $b[j]$