Парки фонтанов

В парке неподалеку находятся n **фонтанов**, пронумерованных от 0 до $n-1$. Будем считать, что фонтаны представляют собой точки на плоскости. А именно, фонтан с номером i ($0 \leq i \leq n-1$) находится в точке $(x[i], y[i])$, где $x[i]$ и $y[i]$ – **четные целые числа**. Все фонтаны находятся в различных точках.

Архитектора Тимати наняли для того, чтобы проложить несколько **дорожек**, а также разместить по одной **лавочке** на каждой из дорожек. Каждая дорожка должна представлять собой **горизонтальный** или **вертикальный** отрезок длины 2, в концах которого находятся два различных фонтана. Дорожки должны быть проложены таким образом, чтобы от любого фонтана до любого другого можно было дойти по дорожкам. Исходно в парке нет дорожек.

На каждой дорожке необходимо разместить **ровно одну** лавочку, которая будет **размещаться на этой дорожке**. Каждая лавочка должна быть расположена в точке (a, b), где a и b – **нечетные целые числа**. Все точки, в которых будут расположены лавочки, должны быть **различными**. Лавочка в точке (a, b) может размещаться на некоторой дорожке, если **оба конца** этой дорожки находятся в множестве $(a-1, b-1)$, $(a-1, b+1)$, $(a+1, b-1)$ и $(a+1, b+1)$. Например, лавочка в точке $(3, 3)$ может размещаться на дорожке, если эта дорожка представляет собой один из следующих отрезков: $(2, 2)$ – $(2, 4)$, $(2, 4)$ – $(4, 4)$, $(4, 4)$ – $(4, 2)$, $(4, 2)$ – $(2, 2)$.

Помогите Тимати выяснить, можно ли проложить дорожки и разместить на них лавочки в соответствии с описанными выше условиями. Если это возможно, необходимо вернуть пример подходящего решения. Если есть несколько возможных решений, можно вернуть любое из них.

Детали реализации

Вам необходимо реализовать следующую функцию:

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

- x, y: два массива длины n. Для каждого i ($0 \leq i \leq n-1$) фонтан i находится в точке $(x[i], y[i])$, где $x[i]$ и $y[i]$ – четные целые числа.
- Если решение существует, то эта функция должна сделать ровно один вызов функции `build` (описанной ниже), чтобы описать решение, а затем функция должна вернуть 1.
- В противном случае функция должна вернуть 0, не вызывая функцию `build`.
- Эта функция будет вызвана ровно один раз.

Ваша реализация должна вызывать следующую функцию, чтобы описать решение – проложенные дорожки и размещенные на них лавочки:

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

- Пусть m обозначает общее проложенных дорожек.
- u, v: два массива длины m, описывающие дорожки, которые необходимо проложить. Эти дорожки пронумерованы от 0 до $m - 1$. Для каждого j ($0 \leq j \leq m - 1$) дорожка j соединяет фонтаны $u[j]$ и $v[j]$. Каждая дорожка должна представлять собой горизонтальный или вертикальный отрезок длины 2. Любые две различные дорожки могут иметь только одну общую точку – один из концов (фонтан). Дорожки должны быть проложены таким образом, чтобы от любого фонтана до любого другого можно было добраться по дорожкам.
- a, b: два массива длины m, описывающие лавочки. Для каждого j ($0 \leq j \leq m - 1$) лавочка, которая размещается на дорожке j, расположена в точке $(a[j], b[j])$. Никакие две различные лавочки не должны размещаться в одной точке.

Примеры

Пример 1

Рассмотрим следующий вызов функции:

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

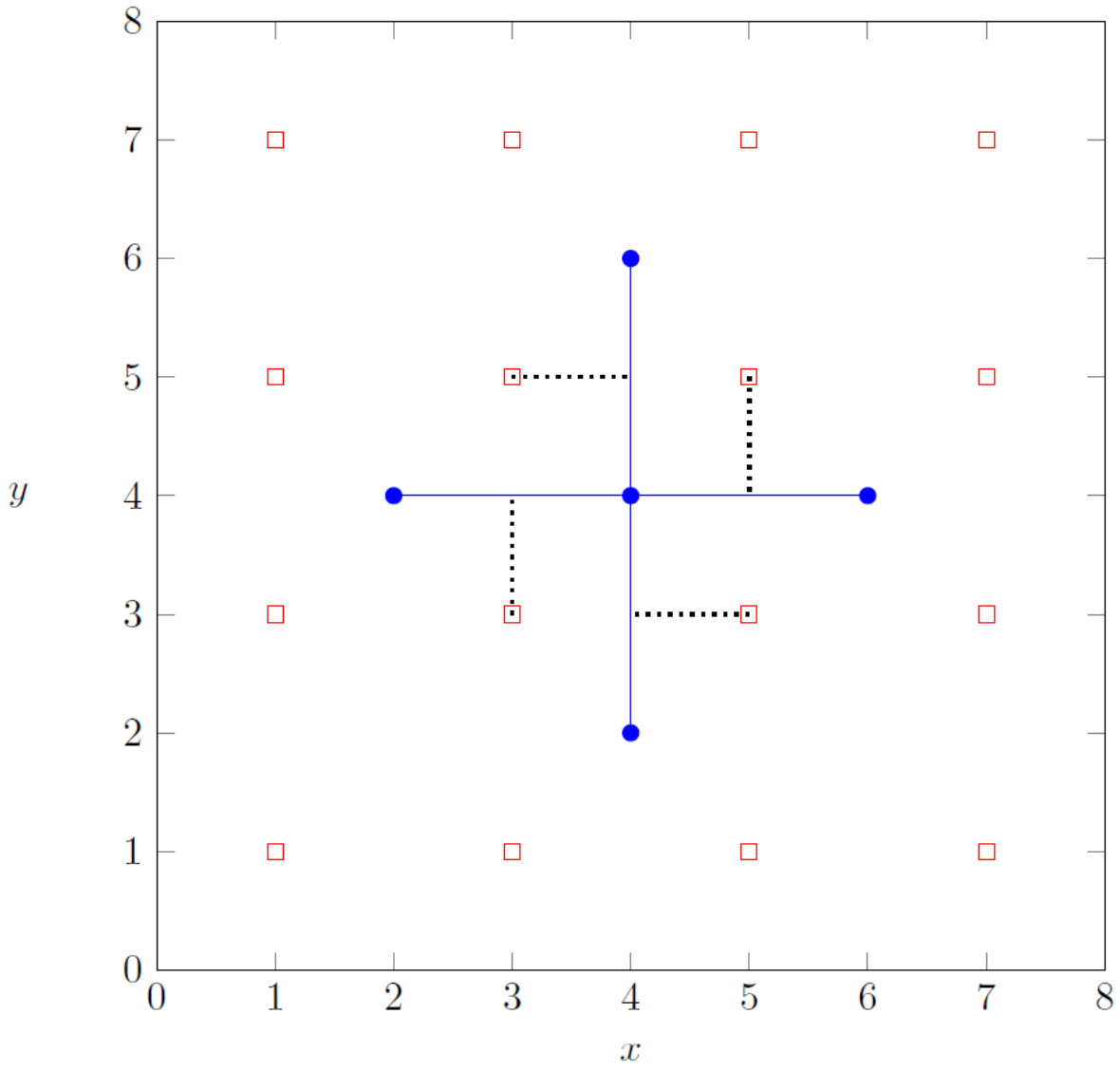
Это означает, что в парке находятся 5 фонтанов:

- фонтан 0 расположен в точке $(4, 4)$,
- фонтан 1 расположен в точке $(4, 6)$,
- фонтан 2 расположен в точке $(6, 4)$,
- фонтан 3 расположен в точке $(4, 2)$,
- фонтан 4 расположен в точке $(2, 4)$.

Можно проложить следующие 4 дорожки, где каждая дорожка соединяет два фонтана, и разместить на них лавочки следующим образом:

Номер дорожки	Фонтаны на концах	Расположение лавочки
0	$0, 2$	$(5, 5)$
1	$0, 1$	$(3, 5)$
2	$3, 0$	$(5, 3)$
3	$4, 0$	$(3, 3)$

Это решение соответствует следующей картинке:



Чтобы описать это решение, `construct_roads` должна сделать следующий вызов:

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

Затем необходимо вернуть значение 1.

Обратите внимание, что в этом примере есть несколько возможных решений, удовлетворяющих ограничениям, любое из них будет принято. Например, также можно описать решение следующим вызовом: `build([1, 2, 3, 4], [0, 0, 0, 0], [5, 5, 3, 3], [5, 3, 3, 5])`, а затем вернуть значение 1.

Пример 2

Рассмотрим следующий вызов функции:

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

Фонтан 0 расположен в точке $(2, 2)$, а фонтан 1 расположен в точке $(4, 6)$. Поскольку проложить дорожки в соответствии с описанными ограничениями нельзя, функция `construct_roads` должна вернуть 0, не вызывая перед этим `build`.

Ограчения

- $1 \leq n \leq 200\,000$
- $2 \leq x[i], y[i] \leq 200\,000$ (для всех $0 \leq i \leq n - 1$)
- $x[i]$ и $y[i]$ – четные целые числа (для всех $0 \leq i \leq n - 1$).
- Никакие два фонтана не расположены в одной точке.

Подзадачи

1. (5 баллов) $x[i] = 2$ (для всех $0 \leq i \leq n - 1$)
2. (10 баллов) $2 \leq x[i] \leq 4$ (для всех $0 \leq i \leq n - 1$)
3. (15 баллов) $2 \leq x[i] \leq 6$ (для всех $0 \leq i \leq n - 1$)
4. (20 баллов) Есть не более одного способа проложить дорожки таким образом, чтобы от любого фонтана до любого другого существовал путь по дорожкам.
5. (20 баллов) Не существует четырех фонтанов, которые находятся в углах квадрата 2×2.
6. (30 баллов) Нет дополнительных ограничений.

Пример грейдера

Пример грейдера читает входные данные в следующем формате:

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

Пример грейдера выводит результат работы решения в следующем формате:

- строка 1: значение, которое вернула функция `construct_roads`

Если функция `construct_roads` вернула 1 и `build(u, v, a, b)` была вызвана, пример грейдера также выводит:

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