Fountain Parks

In a nearby park, there are n **fountains**, labeled from 0 to $n-1$. We model the fountains as points on a two-dimensional plane. Namely, fountain i ($0 \le i \le n-1$) is a point $(x[i], y[i])$ where $x[i]$ and $y[i]$ are **even integers**. The locations of the fountains are all distinct.

Timothy the architect has been hired to plan the construction of some **roads** and the placement of one **bench** per road. A road is a **horizontal** or **vertical** line segment of length 2, whose endpoints are two distinct fountains. The roads should be constructed such that one can travel between any two fountains by moving along roads. Initially, there are no roads in the park.

For each road, **exactly** one bench needs to be placed in the park and **assigned to** (i.e., face) that road. Each bench must be placed at some point (a, b) such that a and b are **odd integers**. The locations of the benches must be all **distinct**. A bench at (a, b) can only be assigned to a road if **both** of the road's endpoints are among $(a-1, b-1)$, $(a-1, b+1)$, $(a+1, b-1)$ and $(a+1, b+1)$. For example, the bench at $(3, 3)$ can only be assigned to a road, which is one of the four line segments $(2, 2)$ – $(2, 4)$, $(2, 4)$ – $(4, 4)$, $(4, 4)$ – $(4, 2)$, $(4, 2)$ – $(2, 2)$.

Help Timothy determine if it is possible to construct roads, and place and assign benches satisfying all conditions given above, and if so, provide him with a feasible solution. If there are multiple feasible solutions that satisfy all conditions, you can report any of them.

Implementation Details

You should implement the following procedure:

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

- x, y: two arrays of length n. For each i ($0 \le i \le n-1$), fountain i is a point $(x[i], y[i])$, where $x[i]$ and $y[i]$ are even integers.
- If a construction is possible, this procedure should make exactly one call to `build` (see below) to report a solution, following which it should return 1.
- Otherwise, the procedure should return 0 without making any calls to `build`.
- This procedure is called exactly once.

Your implementation can call the following procedure to provide a feasible construction of roads and a placement of benches:

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

- Let m be the total number of roads in the construction.

- u, v: two arrays of length m, representing the roads to be constructed. These roads are labeled from 0 to $m-1$. For each j ($0 \leq j \leq m-1$), road j connects fountains $u[j]$ and $v[j]$. Each road must be a horizontal or vertical line segment of length 2. Any two distinct roads can have at most one point in common (a fountain). Once the roads are constructed, it should be possible to travel between any two fountains by moving along roads.
- a, b: two arrays of length m, representing the benches. For each j ($0 \leq j \leq m-1$), a bench is placed at $(a[j], b[j])$, and is assigned to road j. No two distinct benches can have the same location.

Examples

Example 1

Consider the following call:

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

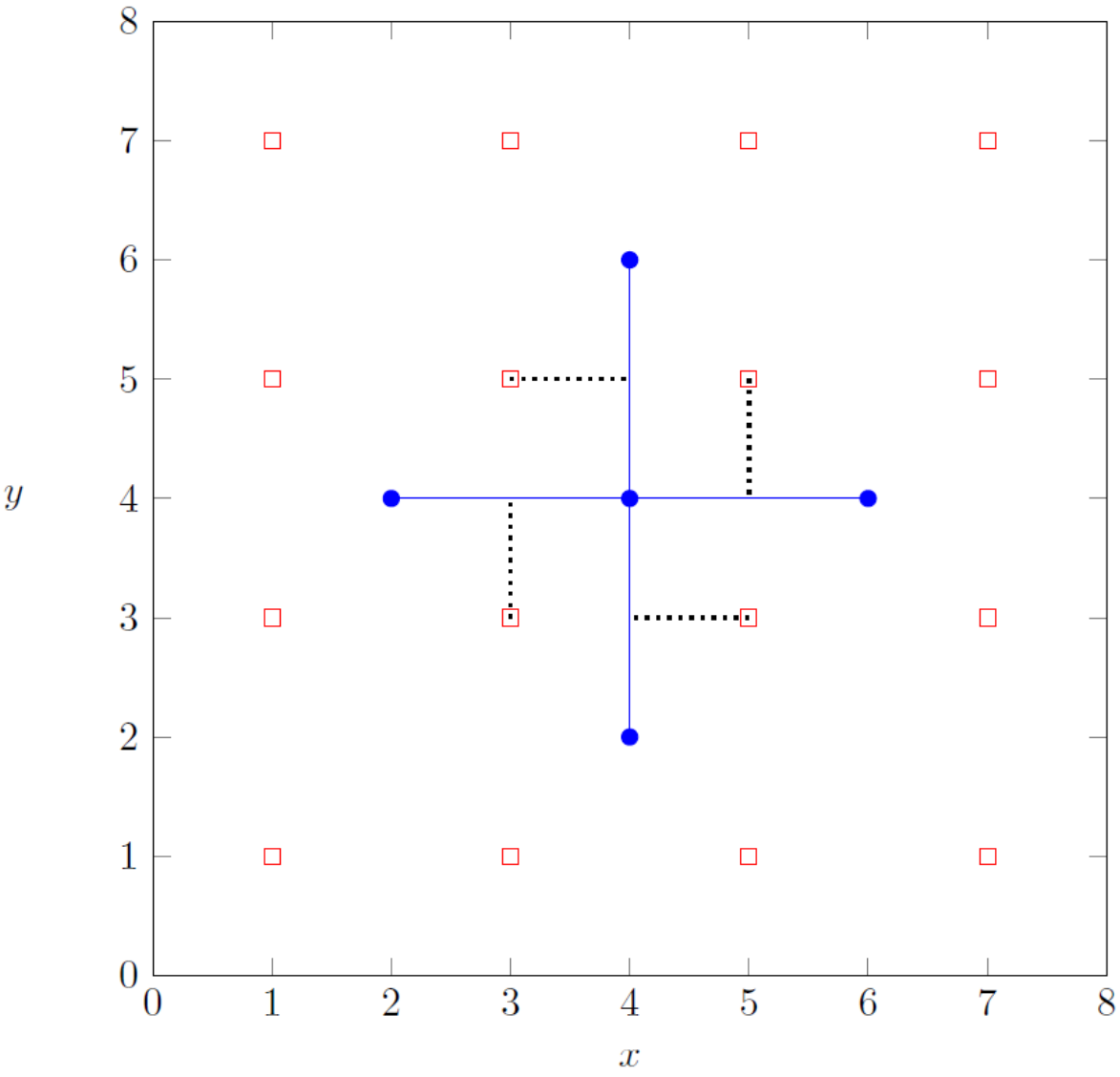
This means that there are 5 fountains:

- fountain 0 is located at $(4, 4)$,
- fountain 1 is located at $(4, 6)$,
- fountain 2 is located at $(6, 4)$,
- fountain 3 is located at $(4, 2)$,
- fountain 4 is located at $(2, 4)$.

It is possible to construct the following 4 roads, where each road connects two fountains, and place the corresponding benches:

Road label	Labels of the fountains the road connects	Location of the assigned bench
0	$0, 2$	$(5, 5)$
1	$0, 1$	$(3, 5)$
2	$3, 0$	$(5, 3)$
3	$4, 0$	$(3, 3)$

This solution corresponds to the following diagram:



To report this solution, `construct_roads` should make the following call:

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

It should then return 1.

Note that in this case, there are multiple solutions that satisfy the requirements, all of which would be considered correct. For example, it is also correct to call `build([1, 2, 3, 4], [0, 0, 0, 0], [5, 5, 3, 3], [5, 3, 3, 5])` and then return 1.

Example 2

Consider the following call:

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

Fountain 0 is located at $(2, 2)$ and fountain 1 is located at $(4, 6)$. Since there is no way to construct roads that satisfy the requirements, `construct_roads` should return 0 without making any

call to `build`.

Constraints

- $1 \leq n \leq 200\,000$
- $2 \leq x[i], y[i] \leq 200\,000$ (for all $0 \leq i \leq n-1$)
- $x[i]$ and $y[i]$ are even integers (for all $0 \leq i \leq n-1$).
- No two fountains have the same location.

Subtasks

1. (5 points) $x[i] = 2$ (for all $0 \leq i \leq n-1$)
2. (10 points) $2 \leq x[i] \leq 4$ (for all $0 \leq i \leq n-1$)
3. (15 points) $2 \leq x[i] \leq 6$ (for all $0 \leq i \leq n-1$)
4. (20 points) There is at most one way of constructing the roads, such that one can travel between any two fountains by moving along roads.
5. (20 points) There do not exist four fountains that form the corners of a 2×2 square.
6. (30 points) No additional constraints.

Sample Grader

The sample grader reads the input in the following format:

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

The output of the sample grader is in the following format:

- line 1: the return value of `construct_roads`

If the return value of `construct_roads` is 1 and `build(u, v, a, b)` is called, the grader then additionally prints:

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