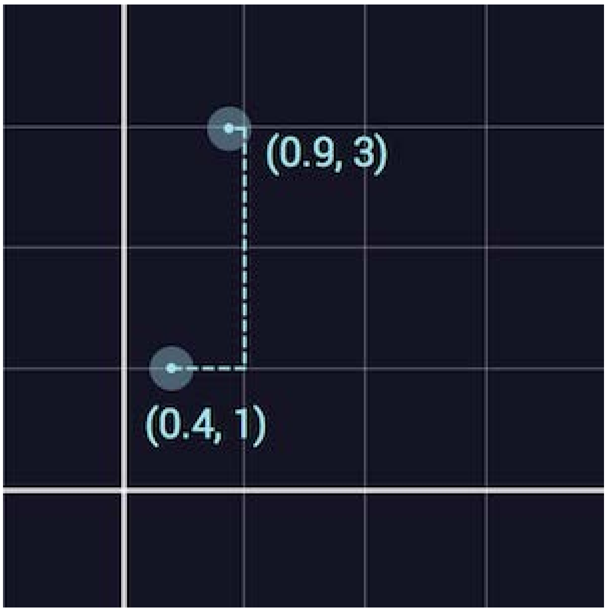
Consider a city where the streets are perfectly laid out to form an infinite square grid. In this city finding the shortest path between two given points (an origin and a destination) is much easier than in other more complex cities. As a new Uber developer, you are tasked to create an algorithm that does this calculation.

Given user's departure and destination coordinates, each of them located on some street, find the length of the shortest route between them assuming that cars can only move along the streets. Each street can be represented as a straight line defined by the `x = n` or `y = n` formula, where `n` is an integer.

**Example**

For `departure = [0.4, 1]` and `destination = [0.9, 3]`, the output should be `perfectCity(departure, destination) = 2.7`.

`0.6 + 2 + 0.1 = 2.7`, which is the answer.



**Input/Output**

- **[time limit] 4000ms (py)**

- **[input] array.float departure**

  An array `[x, y]` of `x` and `y` coordinates. It is guaranteed that at least one coordinate is integer.

  *Constraints:*

  `0.0 ≤ departure[i] ≤ 10.0`.

- **[input] array.float destination**

  An array `[x, y]` of `x` and `y` coordinates. It is guaranteed that at least one coordinate is integer.

  *Constraints:*

  `0.0 ≤ destination[i] ≤ 10.0`.

- **[output] float**

  The shorted distance between two points along the streets.