G: Zoning Milan

The Head of Architecture at Polimi is asking for your help. For some things you just need to be an engineer. Given a registry of all houses in Milan, you should find out the minimum size of an axis-aligned square zone such that every house in a range of addresses lies in that zone or on its border.

The zoning is a bit lenient, so you can ignore any one house from the range to make the zone smaller. The addresses are given as integers from 1 to N.

Each "zoning request" is given as a consecutive range of houses. A valid zone is the smallest axis-aligned square that contains all of the points in the range, **ignoring at most one.**

Given the (x, y) locations of houses in Milan, and a list of zoning requests, you must figure out for each request: what is the length of a **side** of the smallest axis-aligned square zone that contains all of the houses in the zoning request, possibly ignoring **one** house?

Input

The input will begin with a line containing two integers N and Q, where N is the number of houses, and Q is the number of zoning requests. The next n lines will each contain two integers, x and y, which are the (x, y) coordinates of a house in Milan. The address of this house corresponds with the order in the input. The first house has address 1, the second house has address 2, and so on. No two houses will be at the same location. The next q lines will contain two integers a and b, which represent a zoning request for houses with addresses in the range $[a..b]$ inclusive.

Output

You should output Q lines. On each line print the answer to one of the zoning requests, in order. Each answer will be: the side length of the smallest axis-aligned square that contains all of the points of houses with those addresses, if at most one house can be ignored.

Constraints

- $1 \leq N, Q \leq 10^5$.
- $-10^9 \leq x_i, y_i \leq 10^9$, for $1 \leq i \leq n$
- $1 \leq a_i < b_i \leq n$, for $1 \leq i \leq q$

Scoring

Your program will be tested against several testcases, and will be considered **correct** only if it will solve all of them correctly.

Examples

input	output
3 2 1 0 0 1 1000 1 1 3 2 3	1 0
4 2 0 0 1000 1000 300 300 1 1 1 3 2 4	300 299