Delivery Zone Optimization

📜 Problem Description

A logistics company is working on optimizing its delivery zones to ensure efficient and timely delivery of packages. Each delivery zone is represented as an interval (a[i], b[i]) indicating the inclusive range of house numbers it covers. These delivery zones can overlap.

You are given a list of **n** delivery zones and a maximum allowed length, **k**. Your task is to add exactly **one new delivery zone (a, b)**, with the constraint that its length (b - a) must be less than or equal to **k**.

The goal is to add this new zone in such a way that it **minimizes the number of disconnected sets of delivery zones**.

🔌 Defining "Connected"

A set of delivery zones (a[1], b[1]), (a[2], b[2]), ..., (a[n], b[n]) is considered **connected** if every house number in the range (min(a[1], ..., a[n]), max(b[1], ..., b[n])) is covered by at least one of the delivery zones in the set.

Examples of Connectivity

- **Connected:** The set [(1,2), (2,3), (1,5)] is connected. The overall range is (1, 5), and every house number from 1 to 5 is covered by at least one of those intervals.
- **Disconnected:** The set [(2,2), (3,4)] is not connected. The intervals do not overlap, creating a gap.

Note: The input arrays a and b are 1-based.

📝 Example

Consider the following delivery zones and a k value:

- **Zones:** [(1,5), (2,4), (6,6), (7,14), (16,19)]
- **k:** 2

If you add a new delivery zone of **(5, 7)** (length 2, which is <= k), you can merge the first three zones with the fourth, resulting in two connected sets:

1. [(1,5), (2,4), (5,7), (6,6), (7,14)]
2. [(16,19)]

This gives a total of **2** connected sets.

Alternatively, if you add a new delivery zone of **(14, 16)**, you would end up with three connected sets:

1. [(1,5), (2,4)]
2. [(6,6)]
3. [(7,14), (14,16), (16,19)]

The optimal solution is to add the delivery zone **(5, 7)**, which results in a minimum of **2** connected sets.

📋 Function Signature

You need to implement the following function:

```
public static int minimumSets(List<Integer> a, List<Integer> b, int k)
```

Parameters:

- List<Integer> a: A list containing the starting house numbers for each delivery zone.
- List<Integer> b: A list containing the ending house numbers for each delivery zone.
- int k: The maximum allowed length for the new delivery zone.

Returns:

- int: The minimum number of connected delivery zones achievable after adding exactly one new zone.