DSA

Name ……. ID ……

(This problem is carefully selected for you from 6605xxxx)

Problem Statement:

You are given **n light beams** in a circular room. Each beam is described by two angles in degrees: a **starting angle** and an **ending angle**, both in the range [0, 360). Each beam lights an arc of the circle in a **clockwise direction**.

For each beam [start, end]:

If start < end, the beam lights the arc from start to end.

If start > end, the beam **wraps around**: it lights from start to 360, then from 0 to end (i.e., from start to end + 360).

If start == end, the beam lights the entire circle (360°).

Relaxing the problem:

It is **guaranteed** that **no other beam overlaps or touches the wrap-around portion** of any beam. That is, if a beam wraps around (i.e., start > end), then no other beam will cover or touch any part of the arc from start to end + 360. This ensures wrap-around beams are **isolated** and do not need complex merging with other beams.

Your Task: write

`List<int[]> merged = mergeBeams(beams);` Merge all overlapping or contiguous beams into the **smallest number of non-overlapping beams**, preserving the total illuminated area.

Return each merged beam as [start, end] where:

Angles are in [0, 360)

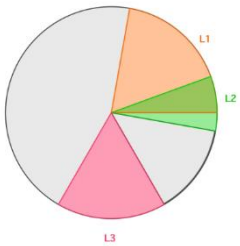
Wrap-around beams may be represented as start > end

Full-circle beams are represented as [a, a] for any angle a

Input: n and 2 * n values for start, end representing a beam as [start, end]

3

10 90 70 100 150 210



Output: A list of merged beams, each also represented as [start, end]

[10, 100]

[150, 210]

DSA

Name ……. ID ……

```
static void demo_1() {
    int n = 3;
    int [] data = {10, 90, 70, 100, 150, 210};
    List<int[]> beams = new ArrayList<>();
    for (int i = 0; i < 2*n; i+=2) {
        beams.add(new int[] {data[i],data[i+1]});
    }
    List<int[]> merged = mergeBeams(beams);

    for (int[] beam : merged) {
        System.out.println(Arrays.toString(beam));
    }
}
```

You may use the following guidelines.

List<int[]> beams = Arrays.asList(

new int[]{70, 100}, new int[]{150, 180}, new int[]{160, 185},

new int[]{350, 60} // wrap-around

);

Step 1: Normalize Beams to linearized Form

i.e. linearized = { {70,100}, {150,180}, {160,185}, {350,420} }

Original Beam	Linearized Beam
[70, 100]	[70, 100]
[150, 180]	[150, 180]
[160, 185]	[160, 185]
[350, 60]	[350, 420]

Step 2: Sort intervals by start angle

i.e. sorted = {{70, 100},{150,180},{160,185},{350,420}}

Step 3: Merge overlapping intervals

*** Your real task ***

i.e. merged = {{70,100},{150,185},{350,420}}

Step 4: Convert Back to Circular Format then return

i.e. result = {{70,100},{150,185},{350,60}}

submit: Lab0_XXYYYY.java which contains mergeBeams() due: TBA