Suppose we have n axis-parallel (that is, the sides are parallel to x and y axes) rectangles. We would like to find the boundary of the union of the interiors of these rectangles. Refer to Figure 1 for a pictorial example.

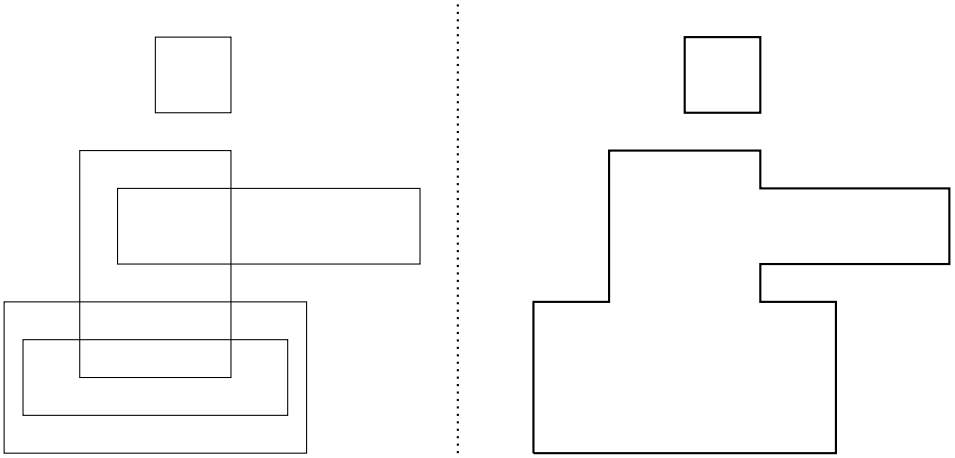

Figure 1: An pictorial example of input (on the left) and output (on the right).

You assume the following:

- ▷ No line segment is a part of a boundary of two rectangles.
- ▷ There are infinitely many vertical lines that pass through the interior of all the rectangles.

Each rectangle is specified by the coordinate of the bottom left corner, length along x-axis, and length along y-axis. The output is specified as the sequence of corner points of the contour. In the example of Figure 1, the input rectangles are specified by the list $\{0, 0, 4, 2\}, \{1, 1, 2, 3\}, \{1.5, 2.5, 4, 1\}, \{0.25, 0.5, 3.5, 1\}, \{2, 4.5, 1, 1\}$. The output is specified by $\{0, 0\}, \{0, 2\}, \{1, 2\}, \{1, 4\}, \{3, 4\}, \{3, 3.5\}, \{5.5, 3.5\}, \{5.5, 2.5\}, \{3, 2.5\}\{3, 2\}, \{4, 2\}, \{4, 0\}, \{0, 0\}, \{2, 4.5\}, \{2, 5.5\}, \{3, 5.5\}, \{3, 4.5\}, \{2, 4.5\}$.

Part I: Compute a Vertical Piercing Line

Write a function which takes all the rectangles as input and returns a vertical line which passes through the interior of all the input rectangles. Define and use an appropriate function prototype. Your algorithm should run in $\mathcal{O}(n)$ time.

Part II: Compute the Boundary

Here is a high level idea of an algorithm for the problem. "Cut the 2-D plane" along the piercing vertical line found above thereby dividing the original problem into two "simpler" sub-problems (why simpler?). Solve each of the sub-problems using a divide and conquer methodology and combine the solutions to obtain the final output. Your algorithm should run in $\mathcal{O}(n \log n)$ time where n is the number of input rectangles.

main()

1. Read n from the user.
2. Dynamically allocate space to store n rectangles using malloc/calloc/new
3. Compute the boundary and output

Submit a single .c or .cpp file. Your code should get compiled properly by gcc or g++ compiler.

Sample Output

```
Write n: 4
0 0 4 2
1 1 2 3
1.5 2.5 4 1
2 4.5 1 1

Boundary: (0,0),(0,2),(1,2),(1,4),(3,4),(3,3.5),(5.5,3.5),(5.5,2.5),(3,2.5),(3,2),(4,2),
(4,0),(0,0),(2,4.5),(2,5.5),(3,5.5),(3,4.5),(2,4.5)
```