ASSIGNMENT-6
PROGRAMMING IN JAVA LABORATORY (CAP680)

Submitted to: **Jaswinder Singh**

Submitted by: **Vyshnav | 12205220 | 41(B) | D2213**

DATE: -**26/03/2023**

Codehs link: - https://codehs.com/share/id/assignment-6-g2-iPcpF0/run

EVEN

EVEN

{Using Sandbox only}
You are given N intervals, the I'th of them being [Ai, Bi], where Ai and Bi are positive integers. Let the union of all these intervals be S. It is easy to see that S can be uniquely represented as an union of disjoint closed intervals. Your task is to find the sum of the lengths of the disjoint closed intervals that comprises S. For example, if you are given the intervals: [1, 3], [2, 4], [5, 7] and [7, 8], then S can be uniquely represented as the union of disjoint intervals [1, 4] and [5, 8]. In this case, the answer will be 6, as (4 - 1) + (8 - 5) = 6.

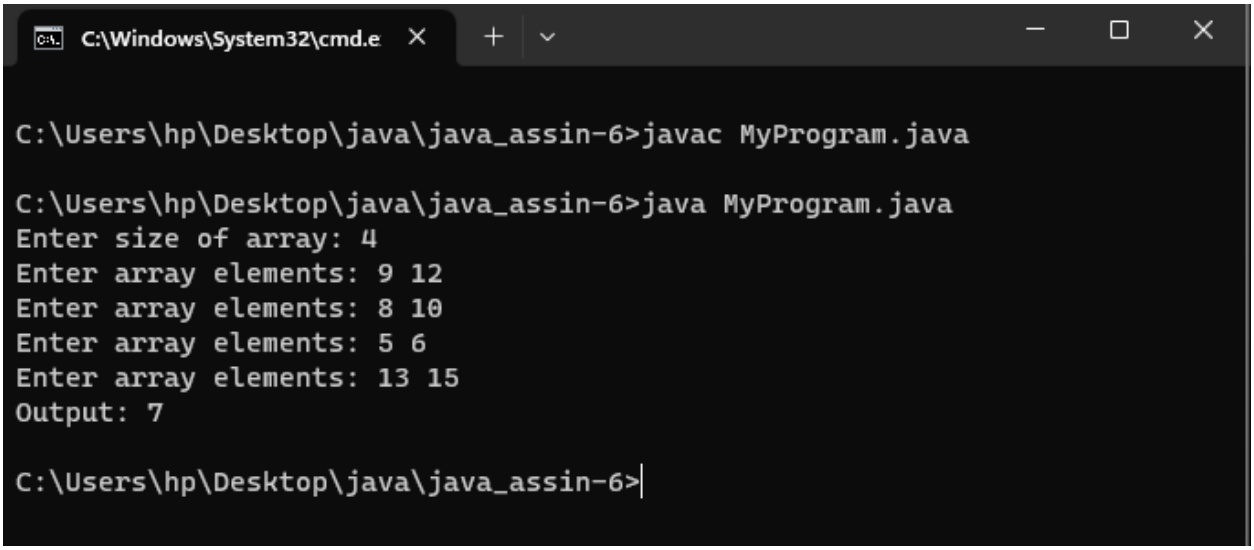
Input
The first line of the input consists of a single integer N – the number of intervals.
Then N lines follow, the it line containing two space-separated integers Ai and Bi.

Constraints:
$1 \leq N \leq 104$
$1 \leq Ai < Bi \leq 2 \times 105$

Output
Print a single integer, the sum of the lengths of the disjoint intervals of S.

Example
Sample Input 1:
3
1 3
3 4
6 7

Sample Output 1:
4



CODE USED

```
import java.util.*;

public class MyProgram {

   public static void main(String[] args) {

      Scanner sc = new Scanner(System.in);

      System.out.print("Enter size of array: ");

      int n = sc.nextInt();

      int[][] intervals = new int[n][2];

      for (int i = 0; i < n; i++) {

         System.out.print("Enter array elements: ");

         intervals[i][0] = sc.nextInt();

         intervals[i][1] = sc.nextInt();

      }

      Arrays.sort(intervals, (a, b) -> a[0] - b[0]);

      int start = intervals[0][0];
```

```
        int end = intervals[0][1];
        int len = 0;
        for (int i = 1; i < n; i++) {
            if (intervals[i][0] > end) {
                len += end - start;
                start = intervals[i][0];
                end = intervals[i][1];
            } else {
                end = Math.max(end, intervals[i][1]);
            }
        }
        len += end - start;
        System.out.print("Output: ");
        System.out.println(len);
    }
}
```

----------End of project----------