UNITED INTERNATIONAL UNIVERSITY
Department of Computer Science and Engineering (CSE)

Course Title: Data Structure & Algorithm Lab Lab II **Course Code: CSE2218**

Trimester & Year: Fall 2021 **Section: D** **Credit Hours: 1.0** **AZ**

ASSIGNMENT 02: Greedy Algorithm

Q1: Kruskal Algorithm Implementaion

Implement Kruskal's algorithm using the following pseudocode as discussed in the Class Lecture and show its simulation (Deposit the simulation as a separate pdf named *1.pdf*)

MST-KRUSKAL(G, w)

$A = \emptyset$
for each vertex $v \in G.V$
    MAKE-SET(v)
sort the edges of $G.E$ into nondecreasing order by weight w
for each edge $(u, v) \in G.E$, taken in nondecreasing order by weight
    if FIND-SET$(u) \neq$ FIND-SET(v)
        $A = A \cup \{(u, v)\}$
        UNION(u, v)
return A

Q2: Feed The Children

Assume you are an awesome parent and want to give your children some cookies. But, you should give each child at most one cookie.

Each child i has a greed factor g[i], which is the minimum size of a cookie that the child will be content with; and each cookie j has a size s[j]. If s[j] >= g[i], we can assign the cookie j to the child i, and the child i will be content. Your goal is to maximize the number of your content children and output the maximum number.

Example 1:

Input: `g = [1,2,3], s = [1,1]`	**Output:** `1`

Explanation: You have 3 children and 2 cookies. The greed factors of 3 children are 1, 2, 3.

And even though you have 2 cookies, since their size is both 1, you could only make the child whose greed factor is 1 content.

You need to output 1.

Example 2:

Input: `g = [1,2], s = [1,2,3]`	**Output:** `2`

Explanation: You have 2 children and 3 cookies. The greed factors of 2 children are 1, 2.

You have 3 cookies and their sizes are big enough to gratify all of the children,

You need to output 2.

Constraints:

- `1 <= g.length <= 3 * 104`
- `0 <= s.length <= 3 * 104`
- `1 <= g[i], s[j] <= 231 - 1`

UNITED INTERNATIONAL UNIVERSITY
Department of Computer Science and Engineering (CSE)

Q3: Course Selection Problem

X-land has a very famous university. The university offers N courses. Each course runs for some consecutive range of days. You are given starting and ending days of the i^{th} course by $start_i$ and end_i, respectively.
Sam wanted to enrol himself in the university. But he is not sure about the exact time for which he wants to study. Though he has Q such tentative plans in his mind. Each plan consists of a start date $plan\_start_j$ and an end date $plan\_end_j$.

Sam wants your help in finding out the maximum number of courses he can complete during each of his plans. Note that at a time, Sam can not handle multiple courses, i.e. he can attend at most one course during a day. Also, a course will be considered completed only if Sam attends all the classes of the course.

Input

There is a single test case.

The first line of the input contains two space-separated integers N and Q denoting the number of courses the university offers and the number of plans Sam has in mind, respectively.

The i^{th} of the next N lines contains two space-separated integers $start_i$ and end_i denoting the starting and the ending day of the i^{th} course.

The j^{th} of the next Q lines contains two space-separated integers $plan\_start_j$ and $plan\_end_j$, denoting the start and the end day of Sam's plan.

Output

Output **Q** lines - each containing an integer corresponding to the maximum number of the courses Sam can complete in the corresponding planned visit.

Example

Input:	Output:
3 3 1 3 5 6 2 4 1 6 1 3 2 3	2 1 0

UNITED INTERNATIONAL UNIVERSITY
Department of Computer Science and Engineering (CSE)

Q4: Burst The Balloons

There are some spherical balloons taped onto a flat wall that represents the XY-plane. The balloons are represented as a 2D integer array of points where `points[i] = [`x_{start}`, `x_{end}`]` denotes a balloon whose horizontal diameter stretches between x_{start} and x_{end}. You do not know the exact y-coordinates of the balloons.

Arrows can be shot up directly vertically (in the positive y-direction) from different points along the x-axis. A balloon with x_{start} and x_{end} is burst by an arrow shot at x if $x_{start} <= x <= x_{end}$. There is no limit to the number of arrows that can be shot. A shot arrow keeps traveling up infinitely, bursting any balloons in its path.
Given the array points, return the *minimum number of arrows* that must be shot to burst all balloons.

Example 1:

Input: `[[10,16],[2,8],[1,6],[7,12]]`	**Output:** `2`

Explanation: The balloons can be burst by 2 arrows:

- Shoot an arrow at x = 6, bursting the balloons [2,8] and [1,6].

- Shoot an arrow at x = 11, bursting the balloons [10,16] and [7,12].

Example 2:

Input: `[[1,2],[3,4],[5,6],[7,8]]`	**Output:** `4`

Explanation: One arrow needs to be shot for each balloon for a total of 4 arrows.

Constraints:

- $1 <= points.length <= 10^5$
- `points[i].length == 2`
- $-2^{31} <= x_{start} < x_{end} <= 2^{31} - 1$