Cow Crossings

Every day, Farmer John's N cows (1 <= N <= 100,000) cross a road in the middle of his farm. Considering the map of FJ's farm in the 2D plane, the road runs horizontally, with one side of the road described by the line y=0 and the other described by y=1. Cow i crosses the road following a straight path from position (a_i, 0) on one side to position (b_i, 1) on the other side. All the a_i's are distinct, as are all the b_i's, and all of these numbers are integers in the range -1,000,000...1,000,000.

Despite the relative agility of his cows, FJ often worries that pairs of cows whose paths intersect might injure each-other if they collide during crossing. FJ considers a cow to be "safe" if no other cow's path intersects her path. Please help FJ compute the number of safe cows.

INPUT FORMAT:

* Line 1: The number of cows, N.

* Lines 2..1+N: Line i contains the integers a_i and b_i, describing

    the path taken by cow i.

SAMPLE INPUT:

4

-3 4

7 8

10 16

3 9

INPUT DETAILS:

There are 4 cows. Cow 1 follows a straight path from (-3,0) to (4,1), and so on.

OUTPUT FORMAT:

* Line 1: The number of safe cows.

SAMPLE OUTPUT:

2

OUTPUT DETAILS:

The first and third cows each do not intersect any other cows. The second and fourth cows intersect each other.

Milk Scheduling

Farmer John's N cows (1 <= N <= 10,000) are conveniently numbered 1..N. Each cow i takes T(i) units of time to milk. Unfortunately, some cows must be milked before others, owing to the layout of FJ's barn. If cow A must be milked before cow B, then FJ needs to completely finish milking A before he can start milking B.

In order to milk his cows as quickly as possible, FJ has hired a large number of farmhands to help with the task -- enough to milk any number of cows at the same time. However, even though cows can be milked at the same time, there is a limit to how quickly the entire process can proceed due to the constraints requiring certain cows to be milked before others. Please help FJ compute the minimum total time the milking process must take.

INPUT FORMAT:

* Line 1: Two space-separated integers: N (the number of cows) and M

(the number of milking constraints; 1 <= M <= 50,000).

* Lines 2..1+N: Line i+1 contains the value of T(i) (1 <= T(i) <= 100,000).

* Lines 2+N..1+N+M: Each line contains two space-separated integers A

and B, indicating that cow A must be fully milked before one

can start milking cow B. These constraints will never form

a cycle, so a solution is always possible.

SAMPLE INPUT:

3 1

10

5

6

3 2

INPUT DETAILS:

There are 3 cows. The time required to milk each cow is 10, 5, and 6, respectively. Cow 3 must be fully milked before we can start milking cow 2.

OUTPUT FORMAT:

* Line 1: The minimum amount of time required to milk all cows.

SAMPLE OUTPUT:

11

OUTPUT DETAILS:

Cows 1 and 3 can initially be milked at the same time. When cow 3 is finished with milking, cow 2 can then begin. All cows are finished milking after 11 units of time have elapsed.

Taxi

Bessie is running a taxi service for the other cows on the farm. The cows have been gathering at different locations along a fence of length M (1 <= M <= 1,000,000,000). Unfortunately, they have grown bored with their current locations and each wish to go somewhere else along the fence. Bessie must pick up each of her friends at their starting positions and drive them to their destinations. Bessie's car is small so she can only transport one cow in her car at a time. Cows can enter and exit the car instantaneously.

To save gas, Bessie would like to minimize the amount she has to drive. Given the starting and ending positions of each of the N cows (1 <= N <= 100,000), determine the least amount of driving Bessie has to do. Bessie realizes that to save the most gas she may need to occasionally drop a cow off at a position other than her destination.

Bessie starts at the leftmost point of the fence, position 0, and must finish her journey at the rightmost point on the fence, position M.

INPUT FORMAT:

* Line 1: N and M separated by a space.

* Lines 2..1+N: The (i+1)th line contains two space separated

integers, s_i and t_i (0 <= s_i, t_i <= M), indicating the

starting position and destination position of the ith cow.

SAMPLE INPUT:

2 10

0 9

6 5

INPUT DETAILS:

There are two cows waiting to be transported along a fence of length 10. The first cow wants to go from position 0 (where Bessie starts) to position 9. The second cow wishes to go from position 6 to position 5.

OUTPUT FORMAT:

* Line 1: A single integer indicating the total amount of driving

Bessie must do. Note that the result may not fit into a 32 bit

integer.

SAMPLE OUTPUT:

12

OUTPUT DETAILS:

Bessie picks up the first cow at position 0 and drives to position 6. There she drops off the first cow, delivers the second cow to her destination and returns to pick up the first cow. She drops off the first cow and then drives the remainder of the way to the right side of the fence.

Route Designing

After escaping from the farm, Bessie has decided to start a travel agency along the Amoozon river. There are several tourist sites located on both sides of the river, each with an integer value indicating how interesting the tourist site is.

Tourist sites are connected by routes that cross the river (i.e., there are no routes connecting a site with a site on the same side of the river). Bessie wants to design a tour for her customers and needs your help. A tour is a sequence of tourist sites with adjacent sites connected by a route. In order to best serve her customers she wants to find the route that maximizes the sum of the values associated with each visited site.

However, Bessie may be running several of these tours at the same time. Therefore it's important that no two routes on a tour intersect. Two routes (a <-> x) and (b <-> y) intersect if and only if (a < b and y < x) or (b < a and x < y) or (a = b and x = y).

Help Bessie find the best tour for her agency. Bessie may start and end at any site on either side of the Amoozon.

INPUT FORMAT:

* Line 1: Three space separated integers N (1 <= N <= 40,000), M (1 <=

  M <= 40,000), and R (0 <= R <= 100,000) indicating

  respectively the number of sites on the left side of the

  river, the number of sites on the right side of the river, and

  the number of routes.

* Lines 2..N+1: The (i+1)th line has a single integer, L_i (0 <= L_i

  <= 40,000), indicating the value of the ith tourist site on

  the left side of the river.

* Lines N+2..N+M+1: The (i+N+1)th line has a single integer, R_i (0 <=

  R_i <= 40,000), indicating the value of the ith tourist site

  on the right side of the river.

* Lines N+M+2..N+M+R+1: Each line contains two space separated

  integers I (1 <= I <= N) and J (1 <= J <= M) indicating there

  is a bidirectional route between site I on the left side of the river and site J on the right side of the river.

SAMPLE INPUT:

3 2 4

1

1

5

2

2

1 1

2 1

3 1

2 2

INPUT DETAILS:

There are three sites on the left side of the Amoozon with values 1, 1, and 5. There are two sites on the right side of the Amoozon with values 2 and 2. There are four routes connecting sites on both sides of the river.

OUTPUT FORMAT:

* Line 1: A single integer indicating the maximum sum of values

attainable on a tour.

SAMPLE OUTPUT:

8

OUTPUT DETAILS:

The optimal tour goes from site 1 on the left, site 1 on the right, and ends at site 3 on the left. These respectively have values 1, 2, and 5 giving a total value of the trip of 8.