At Rally Health we use a lot of Scala for our backend code. It's an interesting language because it allows both functional and object-oriented programming. It's also a language that makes it easy to both write new feature code as well as tests on that code. We don't consider things done until they're both feature-complete as well as unit tested.

Picture a complete feature's work as a grid of size *n* by *m* starting at the origin (0,0) and ending at (*n*-1, *m*-1). There are always states of your progress that don't make sense, as well as discoveries you can make while testing that may set you back a bit, but it's always important to continue making progress. Just like Scala, however, there's a lot of ways to get to your end state.

In a language of your choosing, write a program that takes as input the dimensions of the grid, a list of states that you can't get to (designated by a list grid coordinates), and a list of states that will jump you to a different place altogether (designated by pairs of grid coordinates). Your program should output the total number of different ways to arrive at your goal. Each unit of work can **either** get you one test written (a "jump" between jumpable coordinates) **or** one step closer to feature complete.

Example 1 In:

2 2 *// (a 2x2 grid)*
() *// (no "blocked" points)*
() *// (no "jumping" points)*

Example 1 Out:

2 *// (there are two ways to get from (0,0) to (1,1)*

Example 2 In:

5 5 *// (a 5x5 grid)*
((2 1)) *// (one "blocked" point, the coordinate (2,1))*
() *// (no "jumping" points)*

Example 2 Out:

40

Example 3 In:

5 5 *// (a 5x5 grid)*
() *// (no "blocked" points)*
(((2,1),(0,3))) *// the coordinate (2, 1) jumps you to the coordinate (0,3)*

Example 3 Out:

55