In fact, the proof is really straightforward.
You probably apply a `DFS` , from one stone to next connected stone.
You can remove stones in reversed order.
In this way, all stones can be removed but the stone that you start your DFS.

One more step of explanation:
In the view of DFS, a graph is explored in the structure of a tree.
As we discussed previously,
a tree can be removed in topological order,
from leaves to root.

4. Count the number of islands

We call a connected graph as an island.
One island must have at least one stone left.
The maximum stones can be removed = stones number - islands number

The whole problem is transferred to:
What is the number of islands?

You can show all your skills on a DFS implementation,
and solve this problem as a normal one.

5. Unify index

Struggle between rows and cols?
You may duplicate your codes when you try to the same thing on rows and cols.
In fact, no logical difference between col index and rows index.

An easy trick is that, add 10000 to col index.
So we use 0 ~ 9999 for row index and 10000 ~ 19999 for col.

6. Search on the index, not the points

When we search on points,
we alternately change our view on a row and on a col.

We think:
a row index, connect two stones on this row
a col index, connect two stones on this col.

In another view：