**Analysis of the problem "Robinson and crocodiles"**

There is a rectangular field n × m, in each cell of which there is no more than one crocodile. Each crocodile looks in one of four directions (NORTH, SOUTH, WEST, EAST) and can only move in this direction. A crocodile can run away from the field if there are no other crocodiles in its path. It is required to determine the maximum number of crocodiles that can escape.

Note that if several crocodiles can escape at some moment in time, then the order in which they escape does not matter. Let's simulate the process. We will take turns removing crocodiles that can escape.

We will find a set of crocodiles that can initially escape. Initially, only crocodiles that are on the edge of a row or column and look in the right direction can escape. They can be found in time O(nm).

We will consistently remove crocodiles from the found set. When a crocodile runs away, the one following it (in a row or column) gets a chance to escape and can get into our set.

For each crocodile, we will initially remember all its "neighbors" — crocodiles standing on the same row or in the same column through some (possibly zero) number of empty cells.

To store the set, we will use a doubly linked list. Then removing the crocodile from the set and adding its neighbors works in constant time.

The complexity of this algorithm is O (nm).

**Partial solutions**

We will go through the crocodiles, check if they can escape, and remove the escaped ones. If none of them could escape, we stop.

This solution works for O(n2 m2(n+m)) and passes the first group of tests.

Let's construct an oriented graph, the vertices of which will be crocodiles. If one crocodile interferes with another, then we will connect them with an edge. It remains to determine the number of vertices from which some cycle is achievable.

Such a solution works in O(nm(n+m)) time and memory. It passes the second group of tests.