Homework 3

The exercise asks to write a function check that, given a list of moves for the Hanoi Towers with 4 pegs, and an initial configuration, checks that all moves are ok, where a move is ok when:

i) it does not ask to move a disc from an empty peg,
ii) it does never put a disc on top of a smaller disc.
iii) it only uses pegs "a", "b", "c" and "d".

The function check has the following type

check :: [Move] -> (Int, Config) -> ((Report, [Move]),(Int, Config))

where,

data Report = Bad | Ok deriving Show

type Config =[[Int]] -- a configuration is a list containing 4 lists, corresponding to pegs "a","b","c" and "d", each containing some positive numbers from [1..n], where n is the number of discs. So, for instance, for n=4, this [[1,2],[3],[4],[]] is a correct configuration, The initial configuration initConf is [[1,2,3,4],[],[],[]], the final configuration is [[],[1,2,3,4],[],[]].

The type Report is used to give the final result of the check. Obviously Bad indicates that a wrong move was found and Ok that all moves are correct.

Observe the type of the result of check: ((Report, [Move]),(Int, Config)). It consists of 2 pairs:

a) The first pair indicates whether the sequence of moves is Ok/Bad and, in case that the first component is Bad, the second component contains a list with the wrong move that was encountered. The list contains just this move. If the first component is Ok, then the second component is simply the empty list [].
b) The Int in the second pair indicates the number of moves that are executed till the computation stops, either because a wrong move in found or because all moves have been successfully checked. Initially this number is 0. The second component is simply the configuration reached when the computation stops. In case of success will be the configuration reached after the last move is executed, in case of wrong a move, is the (correct) configuration just before the wrong move is considered. In fact the wrong move in not executed.

In main.hs, contained in the moodle, there will be 2 types of calls of your function check:

- in some cases we use a list ms of moves predefined in the main and call: check ms (0, initConf)

- in other cases we call: check (hanoi n "a" "b" "c" "d") (0, initConf) for a couple of values of n.

For the second type of call you should turn in also the Hanoi tower solution with 4 pegs and the test expects that your function uses the minimal number of moves. In case, the correct function is published in the moodle.

You should upload a file hanoi_check.hs that contains, both hanoi and check and all the functions that they need.

IMPORTANT: you should write the function check using as much as possible Functors, Applicatives and Monads and the operators that come along with them.