```
addToList :: [Int] -> Int -> Int -> [Int]
                      index   value

checkList :: [Int] -> Bool

checkRow ::

checkCW ::

checkCCW :: [Int] -> [Int] -> Bool

lotusSolver :: [Int] -> [Int]

recurse :: [Int] -> Int -> [Int]
recurse A index
    | index == 49 = A
    | fst B = recurse (addToList A index snd B) (index+1)
    | otherwise = recurse
        where B = checkAndRecurse A index 1

checkAndRecurse :: [Int] -> Int -> Int -> (Bool, Int)
                   A        index   val
    | val == 8 = (False, 0)
    | checkCaddToList A index val) = (True, val)
    | checkAndRecurse A index (val+1)
      ^ otherwise
```

A[index]