Homework 2
Types for programs and proofs
due 22 September 2014, 13.15

1. Define disjunction and implication of Booleans in Agda using pattern matching!

```
_||_ : Bool -> Bool -> Bool
_=>_ : Bool -> Bool -> Bool
```

2. Define cut-off subtraction $\dot{-}$ in Agda, where $m \dot{-} n = 0$ if $m \leq n$!

3. (a) Define the power function in Agda using pattern matching!

   (b) The recursion operator is a higher order function which takes a base case and a step case and returns a function defined by primitive recursion with that base case and step case. Its definition in Agda is

   ```
   natrec : {A : Set} -> A -> (Nat -> A -> A) -> Nat -> A
   natrec base step zero     = base
   natrec base step (succ n) = step n (rec base step n)
   ```

   Define the power function in terms of `natrec`!

4. (a) Define the append function which concatenates two lists in Agda!

   (b) Define the append function on vectors, so that the type expresses that the dimension of the output vector is the sum of the dimensions of the input vectors. (Hint: beware that the type-checking algorithm normalizes (computes) the type, and is in this case therefore sensitive to the definition of addition. It matters whether addition is defined by recursion on the first or on the second argument.)

5. Define the functions

```
curry : {A B C : Set} -> (A x B  C)  A -> B -> C
uncurry : {A B C : Set}  (A -> B -> C) -> A x B -> C
```

6. (a) Define equality of natural numbers

   ```
   _==_ : Nat -> Nat -> Bool
   ```

   by pattern matching in both arguments.

(b) Define the same function in terms of natrec! Note that this is harder, because natrec only does recursion in one argument at a time.

(c) Prove that `So (m == n)` iff `m = n` where = is the inductively defined identity type given in the lecture.