Prof. Dr. K. Madlener
Dipl.-Inf. P. Michel
Dipl.-Inf. C. Feller

University of Kaiserslautern
Department of Computer Science
AG Grundlagen der Informatik

Exercise Sheet 6: Specification and Verification with Higher-Order Logic (Summer Term 2011)

Date: 26.05.2011

Exercise 1 Inductive Definitions, Lattices and Fixpoints

a) (Prepare!) Define the reflexive, transitive closure of a relation r as inductive set.

b) (Prepare!) Define a function whose least fixpoint is the aforementioned set.

c) (Prepare!) Let L be a complete lattice, $a, b \in L$ and $a \leq b$. Prove that the closed interval $[a, b]$ is a complete lattice.

   Reminder: $[a, b] := \{x.\ a \leq x \leq b\}$

   It is not required that you solve this exercise in Isabelle/HOL.

Exercise 2 Case Study: Inductive Sets and Fixpoint Induction

In the lecture we have seen the inductive definition of the set of even numbers:

```
inductive_set evens :: "nat set" where
  "0 ∈ evens"
| "n ∈ evens ⟹ n + 2 ∈ evens"
```

Using the generated theorem `evens.induct`, we can then prove that all members of the set are indeed even:

```
theorem evens_are_even:
  "∀x ∈ evens. x mod 2 = 0"
```

a) Prove the theorem using the given induction rule.

b) Define a function `evenf` whose fixpoint is the inductive set `evens`, by deriving it from the inductive definition of `evens`:

```
definition evenf :: "??" where
  "evenf M ≡ ??"
```

c) Formulate an analogous theorem for the least fixpoint of `evenf` (i.e. `lfp evenf`), stating that all elements in the set are even.

d) Prove the theorem using fixpoint induction, specifically the theorem `lfp_ordinal_induct`. Do not use automated methods to prove the theorem and make yourself familiar with the *Find Theorems* function of Isabelle/HOL.

Exercise 3 Case Study: Greatest Common Divisor

a) Consider the following implementation of the greatest common divisor function:

```
fun gcd :: "nat => nat => nat" where
"gcd m 0 = m" |
"gcd m n = gcd n (m mod n)"
```

Prove that the function really computes the greatest common divisor of `m` and `n`.

It might be useful to define and prove the following properties of `gcd` first:

- The result of `gcd` divides both arguments.
- Each common divisor divides the result of `gcd`.
- Each divisor of the result of `gcd` is a common divisor.
- The result of `gcd` is not zero if at least one argument is not zero.

Hint: In Isabelle/HOL, the property that a divides b is expressed by: `a dvd b`.

b) Prove the following property of `gcd`: `k * gcd m n = gcd (k * m) (k * n)`.

c) Consider a slightly different implementation of the greatest common divisor function:

```
fun gcd :: "nat => nat => nat" where
"gcd m n = (if n = 0 then m else gcd n (m mod n))"
```

- Prove that this implementation is equivalent to the first one.
- Prove the property of b) for this implementation.

d) Use the main property of a) to define the greatest common divisor non-recursively with the Hilbert-Choice operator (`SOME`), i.e. not using the Euclidean algorithm.

Prove the equivalence of this function to the original `gcd`.