Assignment 4
Introduction to Computational Logic, SS 2011

Prof. Dr. Gert Smolka, Dr. Chad Brown
www.ps.uni-saarland.de/courses/cl-ss11/

Read in the lecture notes: Chapter 3

Note: It is very important to do all the examples in the lecture notes and the exercises below in the system Coq.

Exercise 4.1 Use a *match* to complete the following definition of a predecessor function *pred′* of type $nat \to option\,nat$ such that $pred'\ O = None$ and $pred'\ (S\,n) = Some\,n$. (Here the first arguments to *None* and *Some* are implicit.) Include the "return" part of the match explicitly.

```
Definition pred' : nat -> option nat :=
```

Prove it satisfies the specification.

Exercise 4.2 Complete the following definition so that it declares an addition function. Use a recursive abstraction with a single argument.

```
Definition addf : nat -> nat -> nat :=
```

Prove that *addf* agrees with the addition function *add* defined in Chapter 2.

Exercise 4.3 For the construction of a meaningless term *bogus*, we can also assume a function $push : nat \to nat$ satisfying $push\ x = S\,(push\ x)$ for all x. Write a section that assumes *push* and proves $S\ bogus = bogus$.

Exercise 4.4 Give the typing rule for *fix*.

Exercise 4.5 Assume the inductive definition of *nat* (with member constructors *O* and *S*) is given as in the lecture notes. Compute the normal form of the following terms:

a) $match\ O\ return\ nat\ with\ O \Rightarrow (S\ O) \mid S\ x \Rightarrow x\ end$

b) $match\ S\ O\ return\ nat\ with\ O \Rightarrow (S\ O) \mid S\ x \Rightarrow x\ end$

c) $match\ S\ x\ return\ nat \to nat\ with\ O \Rightarrow S \mid S\ y \Rightarrow \lambda x : nat.\ f\ x\ y\ end$

Exercise 4.6 Assume the inductive definition of *nat* and the following plain definition are given.

```
Definition d : nat -> nat
 := fix f (x : nat) : nat := match x return nat with O => x | S y => S (S (f y)) end.
```

Compute the normal forms of the following terms. Write out each reduction step and note whether it is a β-reduction, δ-reduction, match-reduction or fix-reduction.

a) $d\,O$

b) $d\,(S\,O)$

c) $\lambda z\,{:}\,nat.d\,z$

d) $\lambda z\,{:}\,nat.d\,(S\,z)$

Exercise 4.7 Give a closed type that has no members.

Exercise 4.8 Explain how the operational typing rule **Appop** can be read algorithmically.

Exercise 4.9 Recall the inductive definition of *pair*.

```
Inductive pair (X Y : Type) : Type :=
| P : X -> Y -> pair X Y.
```

For each constructor introduced by this inductive definition, give the parametric arguments and the proper arguments.