Assignment 2
Semantics, WS 2011-2012

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

Read in the lecture notes: Chapter 2

Note: The test for this assignment will be given in the first 15 minutes of the lecture on Wednesday (since Tuesday is a holiday).

Use Coq's predefined types and functions for booleans, naturals, pairs, lists and options. Predefined objects can be inspected with the command Print. Predefined notation can be inspected with the command Locate. Here are two examples:

```
Locate "*".
Print prod.
```

Exercise 2.1 Define a function that swaps the components of pairs and prove $swap(swap\ p) = p$ for all pairs p.

Exercise 2.2 Prove $x * y = iter\ x\ (plus\ y)\ O$ for all numbers x and y.

Exercise 2.3 Define an exponentiation function *power* and prove *power x n = iter n (mult x) (S O)* for all $x, n : nat$.

Exercise 2.4 Prove the following lemmas.

```
Lemma app_asso (X : Type) (xs ys zs : list X) :
app (app xs ys) zs = app xs (app ys zs).

Lemma length_app (X : Type) (xs ys : list X) :
length (app xs ys) = (length xs) + (length ys).

Lemma rev_app (X : Type) (xs ys : list X) :
rev (app xs ys) = app (rev ys) (rev xs).

Lemma rev_rev (X : Type) (xs : list X) :
rev (rev xs) = xs.
```

Exercise 2.5 Here is a tail-recursive function that obtains the length of a list with an accumulator argument.

```
Fixpoint lengthi {X : Type} (xs : list X) (a : nat) :=
match xs with
| nil => a
| cons _ xr => lengthi xr (S a)
end.
```

Proof the following lemmas.

```
Lemma lengthi_length {X : Type} (xs : list X) (a : nat) :
lengthi xs a = (length xs) + a.
```

```
Lemma length_lengthi {X : Type} (xs : list X) :
length xs = lengthi xs O.
```

Exercise 2.6 Define a predecessor function *nat* → *option nat*.

Exercise 2.7 One can define a bijection between *bool* and *fin2*. Show this fact by completing the definitions and proving the lemmas shown below.

```
Definition f (x : bool) : fin 2 :=
Definition g (x : fin 2) : bool :=
Goal forall b : bool, g (f b) = b.
Goal forall x : fin 2, f (g x) = x.
```

Exercise 2.8 Prove

```
Goal forall X:Type, forall x y z:X, x = y -> y = z -> x = z.
```