Foundations of Computer Software: Exercise 3

January 20, 2012

Note: Before doing the following exercise, load the definitions and theorems proved in the previous exercise.

Exercise 3-1

Define the predicate `even` n (which means that n is an even number) as follows.

```
Inductive even: mynat -> Prop :=
  even_b:  (even Z)
| even_i: forall n:mynat, (even n) -> (even (S (S n))).
```

Run the command: `Check even_ind`. What does the output mean?

Exercise 3-2

Prove the following theorem.

```
Theorem even_plus_even_is_even:
  forall m n: mynat, (even m) -> (even n) -> (even (plus m n)).
```

Exercise 3-3

Define the predicate `odd` n, which means that n is an odd number.

Exercise 3-4

Prove the following theorems.

```
Theorem even_oddS:
  forall n:mynat, (even n) -> (odd (S n)).
```

```
Theorem odd_evenS:
```

```
  forall n:mynat, (odd n) -> (even (S n)).

Theorem odd_plus_odd_is_even:
  forall m n: mynat, (odd m) -> (odd n) -> (even (plus m n)).

Theorem odd_plus_even_is_odd:
  forall m n: mynat, (odd m) -> (even n) -> (odd (plus m n)).
```

Exercise 3-5

The following is another definition of even numbers.

```
Definition even2 :=
  fun n:mynat => exists m:mynat, n=(mult Two m).
```

Prove the following theorems.

```
Theorem even_implies_even2:
  forall n:mynat, (even n) -> (even2 n).

Theorem even2_implies_even:
  forall n:mynat, (even2 n) -> (even n).
```

Exercise 3-6

The following is a yet another definition of even and odd numbers, given by simultaneous induction.

```
Inductive even3 : mynat->Prop :=
   even3_b: (even3 Z)
  |  even3_i: forall n:mynat, (odd3 n) -> (even3 (S n))
with odd3: mynat->Prop :=
   odd3_i: forall n:mynat, (even3 n) -> (odd3 (S n)).
```

Prove that `even` and `even3` are equivalent, i.e. that the following theormes hold.

```
Theorem even_implies_even3:
  forall n:mynat, (even n) -> (even3 n).

Theorem even3_implies_even:
  forall n:mynat, (even3 n) -> (even n).
```