Obliczenia i wnioskowanie w systemie Coq
Lista 2, termin: 26-03-2013

Zadanie 1(2p.)
Zdefiniuj w Coqu typ pusty `empty : Set` (tj. taki, że nie istnieje żaden zamknięty term tego typu) tak, by miał co najmniej jeden konstruktor. Udowodnij, że

```
forall x : empty, 2 * 2 = 100.
```

Zdefiniuj typ `unit : Set` posiadający dokładnie jeden element (term zamknięty tego typu), i udowodnij, że

```
forall x y : unit, x = y.
```

Zadanie 2(4p.)
Zdefiniuj typ `num` do reprezentacji numerałów Churcha. Następnie zdefiniuj funkcję konwersji `nat_2_num` typu `nat -> num` oraz funkcje następnika `succ: num -> num`, dodawania i mnożenia `add, mult: num -> num -> num`. Udowodnij, że funkcje `succ, add, mult` są poprawne (zastanów się, co to znaczy).

Zadanie 3(4p.)
Zapoznaj się z definicją typu indukcyjnego `list` oraz definicją funkcji `rev` służącą do odwracania list. Następnie udowodnij, że `rev` jest inwolucją oraz że jest różnowartościowa.

Zadanie 4(3p.)
Napisz funkcję `nth : list A -> nat -> option A` zwracającą n-ty element listy o elementach pewnego typu `A`. W razie, gdy lista ma mniej niż n+1 elementów, funkcja zwraca `None`; w przeciwnym razie funkcja zwraca `Some a`, gdzie `a` jest n-tym elementem listy. Następnie udowodnij własność (nie używając taktyk automatycznych ani lematów bibliotecznych):

```
Lemma nth_in:
forall n l, n < length l -> exists a:A, nth l n = Some a.
```

Zadanie 5(4p.)

(a) Zdefiniuj w Coqu indukcyjny typ `btree` drzew binarnych etykietowanych elementami typu `nat`. Napisz funkcję typu `bool`, która zwraca wartość `true`, jeśli drzewo zawiera etykietę O, a wartość `false` w przeciwnym wypadku.

(b) Zdefiniuj drugi typ indukcyjny `bbtree` dla takich samych drzew, ale w taki sposób, by konstruktor węzła miał tylko dwa argumenty: etykietę (j.w.) oraz funkcję typu `bool -> bbtree`, która zwraca lewe poddrzewo węzła dla argumentu `true` oraz prawe poddrzewo węzła dla argumentu `false`. Jak w punkcie (a), napisz funkcję wyszukującą 0 w tak reprezentowanym drzewie.

(c) Napisz funkcje konwertujące obiekty typu `btree` do typu `bbtree` i odwrotnie.

(d) Niech funkcje konwertujące nazywają się `conv1` i `conv2`. Udowodnij, że funkcje te definiują bijekcję między typami, tj. udowodnij twierdzenia

$$\texttt{forall b} : \texttt{btree}, \texttt{conv2}(\texttt{conv1 b}) = \texttt{b}$$

oraz

$$\texttt{forall b} : \texttt{bbtree}, \texttt{conv1}(\texttt{conv2 b}) = \texttt{b}.$$