Assignment 3

Deadline for solutions: 15.06.2019

Exercise 1 Cassini's Identity (15 Points)

Cassini's identity is the following property of Fibonacci numbers: $F_{n-1}F_{n+1} - F_n^2 = (-1)^n$. The following elegant proof of Cassini's identity due to Donald Knuth is obtained by resorting to matrix theory:

$$F_{n-1}F_{n+1} - F_n^2 = \det \begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{bmatrix} = \det \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n = \left(\det \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \right)^n = (-1)^n.$$

Implement this proof in Agda using the fact that $\det(M * N) = (\det M) * (\det N)$ without a proof*.

It is advisable to proceed according to the following plan.

1. (Re-)Define Fibonacci numbers by using integers instead of naturals, that is, the type of every F_n is $\mathbb{Z}$.

2. Use the results of Assignment 2 to define a type of 2 by 2 integer matrices.

3. Introduce determinants of matrices: $\det \begin{bmatrix} a & b \\ c & d \end{bmatrix} = a * d - c * b$.

4. Show that

$$\det \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n = (-1)^n.$$

by induction over n using the above auxiliary property of determinants, which you need not prove.

5. Show by induction that

$$\begin{bmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}^n.$$

Hint: Induction step amounts to normalization of the target equality by rewriting. In particular, proving the following equations will be useful.

```
*1ℤ : ∀ {n : ℤ} → n *ℤ 1ℤ ≡ n
+0ℤ : ∀ {n : ℤ} → n +ℤ 0ℤ ≡ n
*0ℤ : ∀ {n : ℤ} → n *ℤ 0ℤ ≡ 0ℤ
```

*This is a rare example of a proof not being required. Unless the requirement to do the proof is explicitly waived like this, it is always mandatory. 'Prove' means 'do prove in Agda'; 'show' = prove.

Exercise 2 Automation through Reflection (15 Points)

The point of the present exercise is to build a simplifier for integer expressions, analogously to the simplifier for list expressions from the lecture.

1. Use the following type for representing integer expressions build of zero, one, negation, summation and multiplication correspondingly over the expressions of the form $z\langle n\rangle$ where z is an element of $\mathbb{Z}$ and n is a *priority* parameter for rearranging sums and products by commutativity.

```
data ℤʳ : Set where
  _⟨_⟩ : ℤ → ℕ → ℤʳ
  0ʳ   : ℤʳ
  1ʳ   : ℤʳ
  -ʳ_  : ℤʳ → ℤʳ
  _+ʳ_ : ℤʳ → ℤʳ → ℤʳ
  _×ʳ_ : ℤʳ → ℤʳ → ℤʳ
```

2. Define the *semantic map*

```
ℤ⟦_⟧ : ℤʳ → ℤ
```

sending expressions to the corresponding values in $\mathbb{Z}$ in the expected way (and removing the priorities).

3. Implement a one-step simplifier `ℤ-simp-step` analogous to `𝕃ʳ-simp-step`, performing the following arithmetic simplifications:

$$x + 0 \to x \qquad x + 0 \to x \qquad (x + y) + z \to x + (y + z)$$
$$x \times 1 \to x \qquad x \times 1 \to x \qquad (x \times y) \times z \to x \times (y \times z)$$
$$x \times (y + z) \to x \times y + x \times z \qquad (x + y) \times z \to x \times z + y \times z$$
$$-(x + y) \to (-x) + (-y) \qquad (-x) \times y \to -(x \times y) \qquad x \times (-y) \to -(x \times y)$$
$$-0 \to 0 \qquad -(-x) \to x \qquad z\langle n\rangle + (-z\langle m\rangle) \to 0 \qquad (-z\langle n\rangle) + z\langle m\rangle \to 0$$
$$z\langle n\rangle + ((-z\langle m\rangle) + y) \to y \qquad (-z\langle n\rangle) + (z\langle m\rangle + y) \to y$$

and moreover swapping the arguments in $z_1\langle n\rangle + z_2\langle m\rangle$ and $z_1\langle n\rangle \times z_2\langle m\rangle$ if the priority m is strictly greater than n. Add further rules to ensure that the latter rearrangements remain stable under associativity of $+$ and $\times$.

4. Define a simplification function

```
ℤ-simp : (t : ℤʳ) → ℕ → ℤʳ
```

analogous to `𝕃ʳ-simp` from the lecture, and prove the soundness property

```
ℤ-simp-sound : (t : ℤʳ) (n : ℕ) → ℤ⟦ t ⟧ ≡ ℤ⟦ ℤ-simp t n ⟧
```

5. Prove the property of 2×2 matrices $\det(M * N) = (\det M) * (\det N)$ from Excerise 1 by normlizing the left and the right hand sides using `ℤ-simp`.

Attention: normalization does not yield the same value, for the left and the right hand sides – the resulting expressions still must be further slightly rearranged to obtain an identity.