Semantics of Programming Languages
Exercise Sheet 1

Before beginning to solve the exercises, open a new theory file named Ex01.thy and write the following three lines at the top of this file.

theory Ex01
imports Main
begin

Exercise 1.1 Calculating with natural numbers

Use the value command to turn Isabelle into a fancy calculator and evaluate the following natural number expressions:

"2 + (2::nat)"  "(2::nat) * (5 + 3)"  "(3::nat) * 4 - 2 * (7 + 1)"

Can you explain the last result?

Exercise 1.2 Natural number laws

Formulate and prove the well-known laws of commutativity and associativity for addition of natural numbers.

Exercise 1.3 Counting elements of a list

Define a function which counts the number of occurrences of a particular element in a list.

fun count :: "'a list ⇒ 'a ⇒ nat"

Test your definition of count on some examples and prove that the results are indeed correct.

Prove the following inequality (and additional lemmas, if necessary) about the relation between count and length, the function returning the length of a list.

theorem "count xs x ≤ length xs"
Exercise 1.4  Adding elements to the end of a list

Recall the definition of lists from the lecture. Define a function \texttt{sno}\texttt{c} that appends an element at the right end of a list. Do not use the existing append operator @ for lists.

\begin{verbatim}
fun sno\texttt{c} :: "'a list ⇒ 'a ⇒ 'a list"
\end{verbatim}

Convince yourself on some test cases that your definition of \texttt{sno}\texttt{c} behaves as expected, for example run:

\begin{verbatim}
value "sno\texttt{c} [] c"
\end{verbatim}

Also prove that your test cases are indeed correct, for instance show:

\begin{verbatim}
lemma "sno\texttt{c} [] c = [c]"
\end{verbatim}

Next define a function \texttt{reverse} that reverses the order of elements in a list. (Do not use the existing function \texttt{rev} from the library.) Hint: Define the reverse of \texttt{x \# xs} using the \texttt{sno}\texttt{c} function.

\begin{verbatim}
fun reverse :: "'a list ⇒ 'a list"
\end{verbatim}

Demonstrate that your definition is correct by running some test cases, and proving that those test cases are correct. For example:

\begin{verbatim}
value "reverse [a, b, c]"
lemma "reverse [a, b, c] = [c, b, a]"
\end{verbatim}

Prove the following theorem. Hint: You need to find an additional lemma relating \texttt{reverse} and \texttt{sno}\texttt{c} to prove it.

\begin{verbatim}
theorem "reverse (reverse xs) = xs"
\end{verbatim}

Homework 1  The doubling function

\textit{Submission until Tuesday, October 23, 10:00 am.}

This homework is to be done both with pen-and-paper and with Isabelle.

You will define recursively the function \texttt{double} that takes one number and returns its double. For example, we have \texttt{double(3) = 6}. Below, by “numbers” we mean “natural numbers”.

Your first task is to define \texttt{double(n)} recursively on \texttt{n}—that is to say, define \texttt{double(0)} and then define \texttt{double(Suc(n))} in terms of \texttt{double(n)}. You are not allowed to use addition or multiplication in the definition.

Another task is to prove that \texttt{double(m + n) = double m + double n} for all numbers \texttt{m} and \texttt{n}.

Finally, you have to prove that your recursive definition of \texttt{double} is correct, in that \texttt{double n = n + n} for all numbers \texttt{n}. 

2