

## Exercises for Computability and Complexity, Spring 2018, Sheet 8

*Please return your solutions in the Thursday lecture on April 12*

***For this exercise sheet please prepare your solutions with a text processor and submit a printout!***

**Exercise 1 (easy)** Design a  $\lambda$ -expression **LISTSUM**, which applied to a list whose entries are Church numerals returns the sum of the list elements, and returns 0 if the list is empty.

**Problem 2 (medium)** Design a  $\lambda$ -expression **sortincreasing**, which applied to a list whose entries are Church numerals returns a list of the same length with the same entries, but sorted in ascending order. You may assume that you already have combinators  $<$ ,  $>$ ,  $\leq$ ,  $\geq$ , which when applied to two Church numerals reduce to **true** or **false** in the obvious fashion. Also you may use all the combinators for Boolean logic, list processing and arithmetics introduced in the lecture notes. Example of what your  $\lambda$ -expression should do: **sortincreasing**  $(1::3::2::1::\mathbf{nil}) \rightarrow^* (1::1::2::3::\mathbf{nil})$ . I suggest to lean on bubblesort in your construction. You will find it helpful to lean on a modular programming style, where you first define lambda expressions for useful subroutines, which you then can use in your main function.