

Exercise Sheet 2: Specification and Verification with Higher-Order Logic (Summer Term 2012)

Date: 18.04.2012

Exercise 1 Functions in Isabelle/HOL

Please do not use the append operator `'op @'` or any other predefined functions on lists for this exercise. (You can use `foldl` for `h`) and `i`), if you like.)

- a) (Prepare!) Write a function `swap : 'a * 'b \Rightarrow 'b * 'a`, which swaps the two components of a pair.
- b) (Prepare!) Write a function `listSwap : ('a * 'b) list \Rightarrow ('b * 'a) list`, which swaps all pairs of a list.
- c) Write a function `map : ('a \Rightarrow 'b) \Rightarrow 'a list \Rightarrow 'b list`, which applies a function to all elements of a list.
- d) Write a function `listSwap2 : ('a * 'b) list \Rightarrow ('b * 'a) list`, with the same behavior as `listSwap`, using the `map` function instead of recursion.
- e) (Prepare!) Write a function `appendRight : 'a list \Rightarrow 'a \Rightarrow 'a list`, which appends a single element at the end of a list.
- f) (Prepare!) Write a function `reverse : 'a list \Rightarrow 'a list`, which reverts a list.
- g) Write a function `replace : 'a \Rightarrow 'a \Rightarrow 'a list \Rightarrow 'a list`, where the call `replace x y l` should return a list in which all occurrences of `x` in `l` are replaced with `y`.
- h) Write a function `forall : ('a \Rightarrow bool) \Rightarrow 'a list \Rightarrow bool`, which calculates whether all elements of a list satisfy the given predicate.
- i) Write a function `exists : ('a \Rightarrow bool) \Rightarrow 'a list \Rightarrow bool`, which calculates whether any element of a list satisfies the given predicate.

For additional exercises in functional programming (e.g. in ML), please refer to the slides, exercise sheets and solutions of the lecture "Software-Entwicklung I" from the winter term 2008/09. (In the winter 2009/10 and later we did Haskell in the beginners course.)

You can also find a lot of suggestions for functions on lists in the documentation of the Haskell `Data.List` module <http://haskell.org/ghc/docs/latest/html/libraries/base-4.5.0.0/Data-List.html>.

Please do not hesitate to ask us, if you encounter any problems when implementing such functions.

Exercise 2 Properties of Functions in Isabelle/HOL

a) Prove or disprove the following properties of the functions `appendRight`, `reverse` and `replace`:

- `replace x y (replace x y l) = replace x y l`
- `y ≠ x → replace x z (replace x y l) = replace x y l`
- `replace y z (replace x y l) = replace x z l`
- `reverse (replace x y l) = replace x y (reverse l)`

b) Prove the following properties of the functions `forall` and `exists`:

- `forall (λx. P x ∧ Q x) l = (forall P l ∧ forall Q l)`
- `exists (λx. P x ∨ Q x) l = (exists P l ∨ exists Q l)`
- `exists P (map f l) = exists (P o f) l`
- `forall P (reverse l) = forall P l`
- `exists P (reverse l) = exists P l`

Exercise 3 Insertion Sort in Isabelle/HOL

a) Specify the sorting algorithm “insertion sort” on lists of integers as primitive recursive function

```
insertionSort :: 'a list ⇒ 'a list
```

in Isabelle/HOL. The elements of the resulting list should be in ascending order.

b) Write a predicate

```
sortedPerm :: 'a list ⇒ 'a list ⇒ bool
```

which takes two lists and tests whether the first is a sorted permutation of the second.

c) Prove the implementation of `insertionSort` correct, regarding the predicate `sortedPerm`.