Isabelle/HOL Exercises
Lists

Searching in Lists

Define a function *first_pos* that computes the index of the first element in a list that satisfies a given predicate:

first_pos :: "('a ⇒ bool) ⇒ 'a list ⇒ nat"

The smallest index is *0*. If no element in the list satisfies the predicate, the behaviour of *first_pos* should be as described below.

Verify your definition by computing

- the index of the first number equal to *3* in the list *[1::nat, 3, 5, 3, 1]*,
- the index of the first number greater than *4* in the list *[1::nat, 3, 5, 7]*,
- the index of the first list with more than one element in the list *[[], [1, 2], [3]]*.

Note: Isabelle does not know the operators > and ≥. Use < and ≤ instead.

Prove that *first_pos* returns the length of the list if and only if no element in the list satisfies the given predicate.

Now prove:

lemma *"list_all (λ x. ¬ P x) (take (first_pos P xs) xs)"*

How can *first_pos (λ x. P x ∨ Q x) xs* be computed from *first_pos P xs* and *first_pos Q xs*? Can something similar be said for the conjunction of *P* and *Q*? Prove your statement(s).

Suppose *P* implies *Q*. What can be said about the relation between *first_pos P xs* and *first_pos Q xs*? Prove your statement.

Define a function *count* that counts the number of elements in a list that satisfy a given predicate.

count :: "('a ⇒ bool) ⇒ 'a list ⇒ nat"

Show: The number of elements with a given property stays the same when one reverses a list with *rev*. The proof will require a lemma.

Find and prove a connection between the two functions *filter* and *count*.