Replace, Reverse and Delete

Define a function `replace`, such that `replace x y zs` yields `zs` with every occurrence of `x` replaced by `y`.

consts `replace :: "'a ⇒ 'a ⇒ 'a list ⇒ 'a list"`

Prove or disprove (by counterexample) the following theorems. You may have to prove some lemmas first.

theorem `"rev(replace x y zs) = replace x y (rev zs)"`
theorem `"replace x y (replace u v zs) = replace u v (replace x y zs)"`
theorem `"replace y z (replace x y zs) = replace x z zs"`

Define two functions for removing elements from a list: `del1 x xs` deletes the first occurrence (from the left) of `x` in `xs`, `delall x xs` all of them.

consts `del1   :: "'a ⇒ 'a list ⇒ 'a list"`
       `delall :: "'a ⇒ 'a list ⇒ 'a list"`

Prove or disprove (by counterexample) the following theorems.

theorem `"del1 x (delall x xs) = delall x xs"`
theorem `"delall x (delall x xs) = delall x xs"`
theorem `"delall x (del1 x xs) = delall x xs"`
theorem `"del1 x (del1 y zs) = del1 y (del1 x zs)"`
theorem `"delall x (del1 y zs) = del1 y (delall x zs)"`
theorem `"delall x (delall y zs) = delall y (delall x zs)"`
theorem `"del1 y (replace x y xs) = del1 x xs"`
theorem `"delall y (replace x y xs) = delall x xs"`
theorem `"replace x y (delall x zs) = delall x zs"`
theorem `"replace x y (delall z zs) = delall z (replace x y zs)"`
theorem `"rev(del1 x xs) = del1 x (rev xs)"`
theorem `"rev(delall x xs) = delall x (rev xs)"`