University of Cambridge
Computer Laboratory
Dr Tjark Weber

Easter Term 2010/11
Exercises 1: Solutions
April 29, 2011

Interactive Formal Verification (L21)

1 Replace, Reverse and Delete

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

```
fun replace :: "'a ⇒ 'a ⇒ 'a list ⇒ 'a list" where
  "replace x y []     = []"
| "replace x y (z#zs) = (if z=x then y else z)#(replace x y zs)"
```

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

```
lemma replace_append: "replace x y (xs @ ys) = replace x y xs @ replace x y
ys"
  apply (induct "xs")
  apply auto
done

theorem "rev (replace x y zs) = replace x y (rev zs)"
  apply (induct "zs")
  apply (auto simp add: replace_append)
done

theorem "replace x y (replace u v zs) = replace u v (replace x y zs)"
  quickcheck
oops
```

A possible counterexample: u=0, v=1, x=0, y=-1, zs=[0]

```
theorem "replace y z (replace x y zs) = replace x z zs"
  quickcheck
oops
```

A possible counterexample: x=1, y=0, z=1, zs=[0]

▷ 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.

```
fun del1 :: "'a ⇒ 'a list ⇒ 'a list" where
  "del1 x []     = []"
| "del1 x (y#ys) = (if y=x then ys else y # del1 x ys)"

fun delall :: "'a ⇒ 'a list ⇒ 'a list" where
  "delall x []     = []"
| "delall x (y#ys) = (if y=x then delall x ys else y # delall x ys)"
```

▷ Prove or disprove (by counterexample) the following theorems.

```
theorem "del1 x (delall x xs) = delall x xs"
  apply (induct "xs")
  apply auto
done

theorem "delall x (delall x xs) = delall x xs"
  apply (induct "xs")
  apply auto
done

theorem delall_del1: "delall x (del1 x xs) = delall x xs"
  apply (induct "xs")
  apply auto
done

theorem "del1 x (del1 y zs) = del1 y (del1 x zs)"
  apply (induct "zs")
  apply auto
done

theorem "delall x (del1 y zs) = del1 y (delall x zs)"
  apply (induct "zs")
  apply (auto simp add: delall_del1)
done

theorem "delall x (delall y zs) = delall y (delall x zs)"
  apply (induct "zs")
  apply auto
done

theorem "del1 y (replace x y xs) = del1 x xs"
  quickcheck
```

```
oops
```

A possible counterexample: x=1, xs=[0], y=0

```
theorem "delall y (replace x y xs) = delall x xs"
  quickcheck
oops
```

A possible counterexample: x=1, xs=[0], y=0

```
theorem "replace x y (delall x zs) = delall x zs"
  apply (induct "zs")
  apply auto
done

theorem "replace x y (delall z zs) = delall z (replace x y zs)"
  quickcheck
oops
```

A possible counterexample: x=1, y=0, z=0, zs=[1]

```
theorem "rev (del1 x xs) = del1 x (rev xs)"
  quickcheck
oops
```

A possible counterexample: x=1, xs=[1, 0, 1]

```
lemma delall_append: "delall x (xs @ ys) = delall x xs @ delall x ys"
  apply (induct "xs")
  apply auto
done

theorem "rev (delall x xs) = delall x (rev xs)"
  apply (induct "xs")
  apply (auto simp add: delall_append)
done
```