

Praktikum Spezifikation und Verifikation

1 Funktionen auf Listen

Definieren Sie eine Funktion *occurs*, so dass *occurs x xs* die Anzahl der Vorkommen von *x* in der Liste *xs* ist.

```
occurs :: "'a ⇒ 'a list ⇒ nat"
```

Beweisen oder widerlegen Sie (mit Gegenbeispiel) nun folgende Theoreme. Eventuell müssen Sie dazu erst einige Lemmas beweisen. Im Fall von Widerlegungen, suchen Sie einige sinnvolle Vorbedingungen damit die Theoreme gelten.

```
theorem [simp]: "occurs a (xs @ ys) = occurs a xs + occurs a ys "
```

```
⋮
```

```
theorem "occurs a xs = occurs a (rev xs)"
```

```
⋮
```

```
theorem "occurs a xs ≤ length xs"
```

```
⋮
```

```
theorem "occurs a (replicate n a) = n"
```

```
⋮
```

Definieren Sie eine Funktion *areAll*, so dass *areAll xs x* gilt, genau dann wenn alle Elemente von *xs* gleich *x* sind.

```
areAll :: "'a list ⇒ 'a ⇒ bool"
```

```
theorem "areAll xs a → occurs a xs = length xs"
```

```
⋮
```

```
theorem "occurs a xs = length xs → areAll xs a"
```

```
⋮
```

Definieren Sie zwei Funktionen zum Entfernen von Elementen aus Listen: *del1 x xs* soll das erste Vorkommen (von links) von *x* in *xs* entfernen. *delall x xs* soll alle Vorkommen von *x* in *xs* entfernen.

```
delall :: 'a => 'a list => 'a list"
del1  :: 'a => 'a list => 'a list"
```

theorem "occurs a (delall a xs) = 0"

:

theorem "Suc (occurs a (del1 a xs)) = occurs a xs"

:

Definieren Sie eine Funktion *replace*, so dass *replace x y zs* bedeutet "ersetze überall *x* durch *y* in *zs*".

```
replace :: 'a => 'a => 'a list => 'a list"
```

theorem "occurs a xs = occurs b (replace a b xs)"

:

Mit Hilfe von *occurs*, definieren Sie eine Funktion *remDups*, die alle doppelten Vorkommen aus der gegebenen Liste entfernt.

```
remDups :: 'a list => 'a list"
```

Formulieren und beweisen Sie ein Lemma, das besagt, dass *remDups* keine neuen Elemente in eine Liste einfügt.

Formulieren und beweisen Sie ein Lemma, das besagt, dass *remDups* immer eine Liste mit paarweise verschiedenen Elementen erzeugt (d.h. die Korrektheit von *remDups*).

Mit Hilfe von *occurs*, definieren Sie jetzt eine Funktion *unique*, so dass *unique xs* gilt, genau dann wenn jedes Element in *xs* genau einmal vorkommt.

```
unique :: 'a list => bool"
```

Mit Hilfe von *unique*, formulieren und beweisen Sie die Korrektheit von *remDups*.

▷ Abgabe: 16. April 2003