

Theorie der Programmierung (WS 2003/04)

Übungsblatt 8

Aufgabe 1

Eine alternative Implementierung der Fakultätsfunktion erhält man, indem man eine Hilfsfunktion einführt, die einen zweiten Parameter zum Aufbau des Ergebnisses benutzt:

```
fun fact_2 (x : int, y : int) : int = if x = 0 then y else fact_2 (x - 1, x * y)  
fun fact (x : int) : int = fact_2 (x, 1)
```

Beweisen Sie die totale Korrektheit von *fact*, d.h. zeigen Sie, dass $fact\ n \Downarrow n!$ für alle $n \geq 0$. (Hinweis: Da die Rekursion in die Hilfsfunktion *fact_2* verlagert ist, benötigt man eine Behauptung über *fact_2*, die sich durch Induktion beweisen lässt, und aus der sich dann die gewünschte Behauptung über *fact* ergibt.)

Aufgabe 2

Die Technik aus Aufgabe 1 kann man auch für eine effiziente Implementierung der *reverse*-Funktion auf Listen ausnutzen. Geben Sie eine solche effiziente Implementierung an und beweisen Sie die totale Korrektheit.

Aufgabe 3

Ein bekannter Algorithmus zum schnellen Potenzieren basiert auf der Gleichung $x^n = x^{n \bmod 2} * (x^2)^{n \operatorname{div} 2}$ (Warum gilt die Gleichung?). Definieren Sie eine entsprechende rekursive Funktion $exp : \mathbf{int} * \mathbf{int} \rightarrow \mathbf{int}$, und beweisen Sie deren totale Korrektheit, d.h. zeigen Sie, dass $exp(m, n) \Downarrow m^n$ für alle $m, n \in \mathbf{Int}, n \geq 0$.

Aufgabe 4

Beweisen Sie die totale Korrektheit unserer simultan rekursiven Implementierung von *even* und *odd*:

```
fun even (x : int) : bool = (x = 0) orelse odd (x - 1)  
and odd (x : int) : bool = not (x = 0) andalso even (x - 1)
```