TECHNISCHE UNIVERSITÄT MÜNCHEN
FAKULTÄT FÜR INFORMATIK

WS 11/12

Einführung in die Informatik 2 **2. Merkblatt**

Prof. Dr. Andrey Rybalchenko, M.Sc. Ruslán Ledesma Garza 05.02.2012

$$\frac{V(b) = v}{V \models b \Rightarrow v} \qquad \frac{}{V \models k \Rightarrow k} \qquad \frac{V \models e_1 \Rightarrow v_1 \quad V \models e_2 \Rightarrow v_2}{V \models e_1 \circ e_2 \Rightarrow v_1 \circ v_2}$$

$$\frac{V \models e_1 \Rightarrow \mathit{true} \quad V \models e_2 \Rightarrow v}{V \models \text{if } e_1 \text{ then } e_2 \text{ else } e_3 \Rightarrow v} \qquad \frac{V \models e_1 \Rightarrow \mathit{false} \quad V \models e_3 \Rightarrow v}{V \models \text{if } e_1 \text{ then } e_2 \text{ else } e_3 \Rightarrow v}$$

$$\frac{V \models e_1 \Rightarrow (\text{fun } f\ b = e,\ t,\ V_1) \quad V \models e_2 \Rightarrow v_2 \quad V_1 + [f := (\text{fun } f\ b = e,\ t,\ V_1)] + [b := v_2] \models e \Rightarrow v}{V \models e_1\ e_2 \Rightarrow v}$$

$$\frac{V_1 = (V \text{ eingeschränkt auf } \mathit{FreeIds}(\text{fun } f\ (b\ :\ t_1)\ :\ t_2 = e))}{V \mid\gg \big(\text{fun } f\ (b\ :\ t_1)\ :\ t_2 = e\big) : V + [f := (\text{fun } f\ b = e,\ t_1 \to t_2,\ V_1)]}$$

$$\frac{V_0 \mid\gg d_1 : V_1 \quad \ldots \quad V_{n-1} \mid\gg d_n : V_n}{V_0 \mid\gg d_1\ \ldots\ d_n : V_n} \qquad \frac{V \models e \Rightarrow v}{V \mid\gg \text{val } b = e : V + [b := v]}$$

```
fun map f nil     = nil
  | map f (x::xr) = (f x) :: (map f xr)

map : ('a -> 'b) -> 'a list -> 'b list

fun filter f nil     = nil
  | filter f (x::xr) = if f x then x :: filter f xr
  else filter f xr

filter : ('a -> bool) -> 'a list -> 'a list

fun exists f nil     = false
  | exists f (x::xr) = f x orelse exists f xr

exists : ('a -> bool) -> 'a list -> bool

fun all f nil     = true
  | all f (x::xr) = f x andalso all f xr

all : ('a -> bool) -> 'a list -> bool

fun foldl f s nil     = s
  | foldl f s (x::xr) = foldl f (f(x,s)) xr

foldl : ('a * 'b -> 'b) -> 'b -> 'a list -> 'b

fun foldr f s nil     = s
  | foldr f s (x::xr) = f(x, foldr f s xr)

foldr : ('a * 'b -> 'b) -> 'b -> 'a list -> 'b
```

```
fun length nil     = 0
  | length (x::xr) = 1 + length xr

length : 'a list -> int

explode : string -> char list

implode : char list -> string
```