SI-möte #3, Programkonstruktion och datastrukturer

Elias Castegren & Kristiina Ausmees
elca7381@student.uu.se ‖ krau6498@student.uu.se

17 november 2011

Begrepp

i) Vad menas med en lista i ML *ii*) Hur fungerar listoperatorerna `::` och `@`?

iii) Hur ska en funktionsspecifikation se ut? Förklara skillnaden på för- och eftervillkor?

iv) Vad är en variant? Vad bevisar varianten (om den är korrekt)?

v) Vad menas med en polymorf typ? Vad är speciellt med en likhetstyp?

Övningar

1.
Skriv funktionspecifikationer för följande funktioner (du behöver inte skriva någon kod):

i) `length` *ii*) `String.sub` *iii*) `hd` *iv*) `tl`

2.
Vilka värden har följande uttryck i ML? Ange dessutom värdenas typer.

i) `hd [#"D",#"M",#"V"]` *ii*) `3::2::[2,4,6]`

iii) `tl ((hd (tl [1,2,3,4,5,6]))::[42])` *iv*) `[]::[]`

v) `[("a"^"b"^"c", size "abc")] @ [("", length([]@[]))]`

vi) `if null (tl [3]) then hd ["a", "b", "c"] else "abc"`

3.
Skriv kod till följande funktioner och ange deras typer. Använd inga inbyggda funktioner från List-biblioteket (förutom `::` och `@`).

i) `third(l)` som beräknar det tredje elementet i listan `l` *(använd matchning!)*

ii) `length(l)` som beräknar längden av listan `l`

iii) `addToList(l, n)` som adderar `n` till alla element i heltalslistan `l`

iv) `removeSign(l)` som returnerar flyttalslistan `l` men med alla element omgjorda till motsvarande absolutbelopp

4.
Betrakta nedanstående kod:

```
val a = 5;
fun f(x) =
  let
    val a = a + x
    val x = a + x
    fun f(x) = a + x
  in
    a + x + f(a)
  end;
```

Vad beräknas anropet `f(a)` till? Får man ett annat resultat om man gör samma anrop en gång till?

5.
Skriv en funktion `sortedMerge(a,b)` som tar två heltalslistor sorterade i stigande ordning (förvillkor) och slår ihop dem till en sorterad lista (eftervillkor). Vad har funktionen för typ?

6.
Skriv en funktion `containsDuplicate(l)` (med fullständig funktionsspecifikation) som tar en lista och returnerar `true` om den innehåller en eller flera dubletter, annars `false`. Hur skulle funktionen kunna göras enklare om man bara tillåter sorterade listor?

7.
Funktionerna `explode` och `implode` kan användas för att omvandla värden mellan strängar och `char`-listor:

`explode("Boom")`$\rightarrow$`[#"B", #"o", #"o", #"m"]`,
`implode([#"m", #"o", #"o", #"B"])`$\rightarrow$`"mooB"`

Använd dessa funktioner för att implementera en egen version av den inbyggda funktionen `String.substring` (den kan till exempel heta `mySubstring`). Börja med funktionsspecifikationen! Använd inga inbyggda funktioner från `String`-biblioteket. Funktionen behöver inte kunna hantera ogiltiga värden.

8.
Skriv en funktion `zip(a,b)` som tar listorna $a = [a_1, a_2, \ldots, a_n]$ och $b = [b_1, b_2, \ldots, b_m]$ och returnerar $[(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)]$. Om det blir några element över i någon lista (dvs om $m \neq n$) skall dessa ignoreras.

Skriv sedan en funktion `unzip(l)` som tar en lista $l = [(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)]$ och returnerar $([a_1, a_2, \ldots, a_n], [b_1, b_2, \ldots, b_n])$

(Både zip och unzip finns inbyggda i ML-biblioteket ListPair)

***9.**
Skriv en funktion `shorter(a, b)` som beräknar `true` om listan `a` är kortare än listan `b`. Skriv funktionen utan att använda `length`! ***Utmaning:*** Kan du skriva funktionen utan matchning eller if-satser?

Lycka till!