Programmierung WS 2002 / 03:
Musterlösung zum 13. Übungsblatt

Prof. Dr. Gert Smolka, Dipl.-Inform. Thorsten Brunklaus

Aufgabe 13.1: Referenzen (2) `ref 1 = ref 1` evaluiert zu `false`, da ref jeweils eine neue Zelle erzeugt und Gleichheit auf Zellen als Tokengleichheit definiert ist.
Aufgabe 13.2: Generatoren ($15 = 3 + 3 + 3 + 6$)

(a)
```
val nextSquare =
  let
    val r = ref 0
  in
   fn () => !r * !r before r := !r + 1
  end
```

(b)
```
fun newNextSquare () =
  let
    val r = ref 0
  in
    fn () => !r * !r before r := !r + 1
  end
```

(c)
```
fun newGenerator f =
  let
    val r = ref 0
  in
    fn () => f(!r) before r:= !r+1
  end
```

(d)
```
fun next y xs =
  if List.all (fn x => y mod x <> 0) xs then y
  else next (y+1) xs

fun newNextPrime () =
  let
    val r = ref []
  in
    fn () =>
      case !r of
        nil   => (r:= [2] ; 2)
      | x::xr => let val p = next (x+1) xr
                 in r:= p:: !r ; p
                 end
  end
```

Aufgabe 13.3: Effiziente imperative Schlangen (15)

```
structure Queue  :>
  sig
    type 'a queue
    val queue  : unit → 'a queue
    val insert : 'a * 'a queue → unit
    val head   : 'a queue → 'a         (* Empty *)
    val remove : 'a queue → unit       (* Empty *)
    val empty  : 'a queue → bool
  end
=
struct
datatype 'a ilist = Nil | Cons of 'a * 'a ilist ref

type 'a queue = 'a ilist ref * 'a ilist ref ref

fun queue () = let val a = ref Nil in (a, ref a) end

fun empty (a,b) = (a= !b)

fun insert (x, (_, b)) =
  let val r = ref Nil in !b:=Cons(x,r) ; b:=r end

fun head (ref(Cons(x,_)), _) = x
  | head         _           = raise Empty

fun remove (ref Nil, _) = raise Empty
  | remove (a as ref(Cons(_,r)), b) =
    (a:= !r ; if r= !b then b:=a else ())
end
```

Aufgabe 13.4: Reversieren von Reihungen (6)

```
fun swap a i j = let val x = Array.sub(a,i)
                 in  Array.update(a, i, Array.sub(a, j)) ;
                     Array.update(a, j, x)
                 end

fun reverse a =
  let
    val l = ref 0
    val u = ref (Array.length a - 1)
  in
    while !l < !u do
      (swap a (!l) (!u) ;
       l:= !l+1 ;
       u:= !u-1 )
  end
```

Aufgabe 13.5: Rotieren von Reihungen (5 + 5)

(a)
```
fun rotate' a au i =
  if i<0 then Array.update(a,0,au)
  else (Array.update(a, i+1, Array.sub(a,i)) ;
        rotate' a au (i-1) )

fun rotate a =
  let
    val u = Array.length a - 1
  in
    rotate' a (Array.sub(a,u)) (u-1)
  end
```

(b)
```
fun rotate a =
  let
    val u  = Array.length a - 1
    val au = Array.sub(a, u)
    val i  = ref (u-1)
  in
    while !i>=0 do
      (Array.update(a, !i+1, Array.sub(a,!i)) ;
       i:= !i-1 ) ;
    Array.update(a,0,au)
  end
```

Aufgabe 13.6: Binäre Suche in Reihungen (9)

```
fun member' a x l u =
  if l>u then false
  else let val m = l+ (u-l) div 2
       in case Int.compare(x, Array.sub(a,m)) of
            LESS    => member' a x l (m-1)
          | EQUAL   => true
          | GREATER => member' a x (m+1) u
       end

fun member a x = member' a x 0 (Array.length a - 1)
```

Aufgabe 13.7: Reversieren von imperativen Listen (5)

```
fun reverse' xs      Nil          = xs
  | reverse' xs (ys as Cons(_,r)) = reverse' ys (!r before r:=xs)

fun reverse xs = reverse' Nil xs
```

Aufgabe 13.8: Zyklische imperative Listen (8)

```
fun circle' r k n =
  Cons(k, if k>=n then r else ref(circle' r (k+1) n))

fun circle n = let
                 val r = ref Nil
                 val f = circle' r 1 n
               in
                 r:=f ; f
               end
```

Aufgabe 13.9: Größe von imperativen Listen (10)

```
fun reflist' r rs =
  case !r of
    Nil        => rs
  | Cons(_,r') => if List.exists (fn r'' => r''=r') rs
                  then rs
                  else reflist' r' (r'::rs)

fun reflist Nil          = nil
  | reflist (Cons(_, r)) = reflist' r [r]

fun size xs = length(reflist xs)
```

Laufzeit von `size` ist quadratisch bezüglich der Größe der Liste.

Aufgabe 13.10: Listenreversion mit Schleifen (5)

```
fun reverse xs =
  let
    val ys = ref nil
    val zs = ref xs
  in
    while not (null(!zs)) do
      (ys:= hd(!zs):: !ys ;
       zs:= tl(!zs) ) ;
    !ys
  end
```

Aufgabe 13.11: Statische Semantik von F mit Referenzen (6)

$$\frac{T \vdash e \Rightarrow t}{T \vdash \texttt{ref}\ e \Rightarrow t\ \texttt{ref}} \qquad \frac{T \vdash e \Rightarrow t\ \texttt{ref}}{T \vdash\ !e \Rightarrow t} \qquad \frac{T \vdash e_1 \Rightarrow t\ \texttt{ref} \quad T \vdash e_2 \Rightarrow t}{T \vdash e_1 := e_2 \Rightarrow \texttt{unit}}$$

Aufgabe 13.12: Fakultät in F plus Referenzen (9)

```
!((fn (fak : (int -> int) ref) =>
  fn (f : (int -> int) ref -> unit) =>
    (fn (dummy : unit) => fak) (f fak))
  (ref (fn (x : int) => x))
  (fn fak => fak := (fn n => if n <= 1 then 1 else n * (!fak)(n - 1))))
```