fun pow (x : int, y : int) = (* correct only for y >= 0 *)
  if y = 0
    then 1
    else x * pow(x, y - 1)
fun cube (x : int) = pow(x, 3)
val sixtyfour = cube(4)
val fortytwo = pow(2, 4) + pow(4, 2) + cube(2) + 2

(* pairs *)
fun swap (pr : int * bool) = (#2 pr, #1 pr)
fun sum_two_pairs (pr1 : int * int, pr2 : int * int) = #1 pr1 + #2 pr1 + #1 pr2 + #2 pr2
  (* returning a pair a real pain in Java *)
fun div_mod (x : int, y : int) = (x div y, x mod y)
fun sort_pair (pr : int * int) =
  if (#1 pr) < (#2 pr)
    then pr
    else (#2 pr, #1 pr)
  (* nested pairs *)
val x1 = (7, (true, 9)) (* int * (bool * int) *)
val x2 = #1 (#2 x1) (* bool *)
val x3 = (#2 x1) (* bool * int *)
val x4 = ((3, 5), ((4, 8), (0, 0))) (* (int * int) * ((int * int) * (int * int)) *)

(* Functions taking or producing lists *)
fun sum_list (xs : int list) =
  if null xs
    then 0
    else hd(xs) + sum_list(tl(xs))
fun countdown (x : int) =
  if x = 0
    then []
    else (#1 (hd xs)) :: (firsts(tl xs))
fun seconds (xs : (int * int) list) =
  if null xs
    then []
    else (#2 (hd xs)) :: (seconds(tl xs))
fun sum_pair_list2 (xs : (int * int) list) =
  (sum_list (firsts xs)) + (sum_list (seconds xs))

(* More functions over lists, here lists of pairs of ints *)
fun sum_pair_list (xs : (int * int) list) =
  if null xs
    then 0
    else #1 (hd(xs)) + #2 (hd(xs)) + sum_pair_list(tl(xs))
fun firsts (xs : (int * int) list) =
  if null xs
    then []
    else (#1 (hd xs)) :: (firsts(tl xs))