type cart = real × real
datatype shape = Circle of cart × real (* coordinates and radius *)
| Square of cart × real (* coordinates and side length *)
| Rectangle of cart × real × real (* coordinates and side lengths *)

fun area sh = case sh of
  Circle(_, r) ⇒ 3.14 × r × r
| Square(_, s) ⇒ s × s
| Rectangle(_, w, l) ⇒ w × l

fun area2 (Circle(_, r)) = 3.14 × r × r
| area2 (Square(_, s)) = s × s
| area2 (Rectangle(_, w, l)) = w × l

(* Here's another nice example showing how pattern matching can be readable, concise, and powerful *)
datatype exp = Constant of int
| Negate of exp
| Add of exp × exp
| Multiply of exp × exp

fun eval (Constant i) = i
| eval (Add(e1, e2)) = (eval e1) + (eval e2)
| eval (Negate e1) = ~ (eval e1)
| eval (Multiply(e1, e2)) = (eval e1) × (eval e2)

(* Tail Recursion *)

fun rev1 xs = case xs of
  [] ⇒ []
| x :: xs' ⇒ append((rev1 xs'), [x])

fun rev2 xs = let
  fun aux (acc, i) =
    if i = 0 then acc
    else aux(xs :: acc, (i - 1))
in
  aux([], n)
end

fun inc_all1 xs = case xs of
  [] ⇒ []
| x :: xs ⇒ (x + 1) :: inc_all1 xs

fun inc_all2 xs = let
  fun aux (acc, xs) =
    case xs of
      [] ⇒ rev2 acc (* need to reverse! still constant space!!! *)
    in
      aux([], xs)
end
choose_second(xs', acc)
end

in
choose_first(l, [])
end