```
fun append (xs,ys) =
    if xs=[]
    then ys
    else (hd xs)::append(tl xs,ys)

fun map (f,xs) =
    case xs of
        [] => []
      | x::xs' => (f x)::(map(f,xs'))

val a = map (increment, [4,8,12,16])
val b = map (hd, [[8,6],[7,5],[3,0,9]])
```

Programming Languages

Dan Grossman
2013

Options

Motivating Options

Having **`max`** return 0 for the empty list is really awful

- Could raise an *exception* (future topic)
- Could return a zero-element or one-element list
  - That works but is poor style because the built-in support for *options* expresses this situation directly

Options

- **`t option`** is a type for any type **`t`**
  - (much like **`t list`**, but a different type, not a list)

Building:

- **`NONE`** has type **`'a option`** (much like **`[]`** has type **`'a list`**)
- **`SOME e`** has type **`t option`** if **`e`** has type **`t`** (much like **`e::[]`**)

Accessing:

- **`isSome`** has type **`'a option -> bool`**
- **`valOf`** has type **`'a option -> 'a`** (exception if given **`NONE`**)

Example

```
fun better_max (xs : int list) =
    if null xs
    then NONE
        else
          let val tl_ans = better_max(tl xs)
          in
              if isSome tl_ans
                 andalso valOf tl_ans > hd xs
              then tl_ans
              else SOME (hd xs)
          end
```

```
val better_max = fn : int list -> int option
```

- Nothing wrong with this, but as a matter of style might prefer not to do so much useless "`valOf`" in the recursion

Example variation

```
fun better_max2 (xs : int list) =
    if null xs
    then NONE
        else let (* ok to assume xs nonempty b/c local
  *)
               fun max_nonempty (xs : int list) =
                 if null (tl xs)
                 then hd xs
                     else
                     let val tl_ans = max_nonempty(tl xs)
                     in
                       if hd xs > tl_ans
                       then hd xs
                       else tl_ans
                     end
           in
               SOME (max_nonempty xs)
           end
```