CSE 341 AA: Section 3

Porter Jones
pbjones@cs.washington.edu
Office Hours: Wednesdays 8:30 - 10:30am

SML Library Stuff

http://sml-family.org/Basis/manpages.html

Unnecessary Function Wrapping

Don’t do it!

Example:

```
fn x => size(x) (* just use size!!! *)
```

Double check your code at a later moment/with a clean slate to spot this!

Map

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

Mystery function 1

```
fun mystery1(p1, p2) =
  case p2 of
      [] => []
    | p::p2' => if p1 p
                then p :: mystery1(p1, p2')
                else mystery1(p1, p2')
```

filter

```
fun filter(f, xs) =
  case xs of
      [] => []
    | x::xs' => if f x
                then x :: filter(f, xs')
                else filter(f, xs')
```

Mystery function 2

```
fun mystery2 (p1, p2, p3) =
    case p3 of
        [] => p2
      | p::p3' => mystery2 (p1, p1(p2,p), p3')
```

fold

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

Extra problems

1. Implement a function even_string_total_length that takes a list of strings and returns the total length of all of the even strings in the given list.

2. Implement flat_map using fold

Extra problems

1. Implement a function even_string_total_length that takes a list of strings and returns the total length of all of the even strings in the given list.

   See next slide for a possible answer...

2. Implement flat_map using fold

```
fun flat_map (f, xs) =
    fold (fn (acc, x) => acc @ f x, [], xs)
```

One way to do it, but there are sooooo many!

```
fun even_string_total_length xs =
  let
    fun even_then_length (acc, s) =
      if size s mod 2 = 0
      then acc + size s
      else acc
  in
    fold (even_then_length, 0, xs)
  end
```