Scheme Tutorial Solutions
Fall 2002

Problem Set 3: Higher-order functions

21. Specific \texttt{check-range} function for determining whether a list of temperatures is between 5°C and 95°C inclusive. This function uses the \texttt{foldl} function.

\begin{verbatim}
;; check-range1 : (listof number) \rightarrow boolean
;; determines if all numbers in a list are between 5 and 95
(define (check-range1 temp-list)
  (foldl lambda (temp accum) (and accum (\leq 5 temp) (\leq temp 95)) true temp-list))
\end{verbatim}

General \texttt{check-range} function for determining whether a list of temperatures is between a min and a max value.

\begin{verbatim}
;; check-range : (listof number) number number \rightarrow boolean
;; determines if all numbers in a list are between a specified max and min value
(define (check-range temp-list min max)
  (foldl lambda (temp accum) (and accum (\leq min temp) (\leq temp max)) true temp-list))
\end{verbatim}

22. Function which converts a list of digits to the corresponding number, using the \texttt{foldr} function.

\begin{verbatim}
;; convert : (listof number) \rightarrow number
;; converts a list of digits to the corresponding number
(define (convert list-digits)
  (foldr lambda (digit accum) (+ (* 10 accum) digit)) 0 list-digits))
\end{verbatim}

23. Functions which together compute the average of a list of prices

\begin{verbatim}
;; count : (listof number) \rightarrow number
(define (count lotp)
  (foldl lambda (elem accum) (+ 1 accum)) 0 lotp))

;; sum : (listof number) \rightarrow number
(define (sum lotp)
  (foldl lambda (elem accum) (+ elem accum)) 0 lotp))
\end{verbatim}
24. Function for converting a list of Fahrenheit temperatures to a list of Celsius temperatures using the map function.

;; tempFC: (listof number) → (listof number)
;; Purpose: To convert a list of Fahrenheit temps to a list of Celsius temps
(define (tempFC list-Ftemps)
  (map (lambda (tempF) (* (/ 5 9) (- tempF 32))) list-Ftemps))

25. Function which uses the filter function to remove all toys with a price greater than ua from a list of toy prices.

;; eliminate-exp: (listof number) number → (listof number)
;; removes all values above a user specified value from a list
(define (eliminate-exp lotp ua)
  (filter (lambda (x) (<= x ua)) lotp))

26. Function which creates a function representing the composition of two functions.

;; compose-func: (Y → Z) (X → Y) → (X → Z)
;; creates a function which is the composition of two other functions
(define (compose-func a b)
  (lambda (x) (a (b x))))

27. Two versions, one using foldr and one without, of a function to convert a list of sublists of numbers to a list of numbers.

;; flatten : (listof (listof number)) → (listof number)
;; flattens a list of sublists of numbers into a list of numbers
(define (flatten list-of-lists)
  (cond
   [(empty? list-of-lists) empty]
   [else (append (first list-of-lists) (flatten (rest list-of-lists)))]))
28. Function using \texttt{foldr} to divide a list into sublists which are composed of adjacent equal numbers.

\begin{verbatim}
;; bucket : (listof number) \rightarrow (listof (listof number))
;; divides a list into a list of sublists, where the sublists are
;; composed of adjacent equal numbers in the original list.
(define (bucket alon)
  (foldr (lambda (elem accum)
       (cond
         [(or (empty? accum)
             (not (= (first (first accum)) elem)))
          (cons (cons elem empty) accum)]
         [else (cons (cons elem (first accum)) (rest accum))]))
    empty alon))
\end{verbatim}

29. Function which applies a function \texttt{f} to the name of every person in the \texttt{family-tree}

\begin{verbatim}
;; tree-map: family-tree (string \rightarrow string) \rightarrow family-tree
;; applies the given function to the name of every person in a family tree
(define (tree-map f tree)
  (cases family-tree tree
    [unknown () (unknown)]
    [person (name birth eye mom dad) (person (f name) birth eye
      (tree-map f mom)
      (tree-map f dad))])
\end{verbatim}

30. Function which uses \texttt{tree-map} to add a last name to every person in a \texttt{family-tree}.

\begin{verbatim}
;; add-last-name : family-tree string \rightarrow family-tree
;; appends a last name to the name of every person in a family tree
(define (add-last-name tree last-name)
  (tree-map (lambda (name) (string-append name " " last-name)) tree))
\end{verbatim}