CMSC 330 Quiz 1 Fall 2021 Solutions

Q1. OCaml Typing

Q1.1. Write an OCaml expression of type (int * string list)

(330, [“cmsc”])

Q1.2. Write an OCaml expression of type 'a -> 'a -> 'a

fun x y -> if x = y then x else y

Q1.3. Write an OCaml expression of type ('a -> 'b) -> ('b -> 'c) -> 'a -> 'c

Hint: Recall function composition from math!

fun f g x -> g (f x)

Q2. OCaml Coding

For all problems in this section, you can use the following functions as given:

let rec map f xs = match xs with
  | [] -> []
  | x::xt -> (f x)::(map f xt)

let rec foldl f a xs = match xs with
  | [] -> a
  | x::xt -> foldl f (f a x) xt

let rec foldr f xs a = match xs with
  | [] -> a
  | x::xt -> f x (foldr f xt a)

let sum x = foldl (fun a x -> a + x) 0 x
let length x = foldl (fun a x -> a + 1) 0 x
let avg x = (sum x)/(length x)

Q2.1. First, write a function, first_k: ('a list -> int -> 'a list), that, given a list and a number k, returns the first k numbers in the list. If the length of the list is less than k, then it returns an empty list.

Examples:

first_k [1; 2; 3; 4] 2 = [1; 2]
first_k [1; 2; 3; 4] 5 = []

let first_k lst k =
  if length lst < k then
    []
  else
    foldl (fun a x -> if length a < k then a @ [x] else a) [] lst
Q2.2. Now, write a function, all_averages: (int list list -> int list) that given a list of lists, finds the average of each sublist.

Example:
all_averages [[1; 2]; [2; 3]; [3; 4]] = [1; 2; 3]

let all_averages lst = map avg lst

Q3. OCaml Rewrite

Given the function (and the declarations defined in Q2)

let rec get_even lst = match lst with
| [] -> []
| h::t -> if h mod 2 = 0 then h::(get_even t) else (get_even t)

Rewrite it so that it doesn't use the rec keyword.

let get_even lst = foldr (fun x a -> if x mod 2 = 0 then x::a else a) lst []