CMSC330 Fall 2011 Example Quiz #3

Name

Discussion Time (circle one): 10am 11am 12pm 1pm 2pm 3pm

Do not start this exam until you are told to do so!

Instructions

- You have 25 minutes for this quiz.
- This is a closed book exam. No notes or other aids are allowed.
- Answer essay questions concisely using 2-3 sentences. Longer answers are not necessary and a penalty may be applied.
- For partial credit, show all of your work and clearly indicate your answers.
- Write neatly. Credit cannot be given for illegible answers.

1. (16 pts) OCaml Types and Type Inference
   a. (2 pts each) Give the type of the following OCaml expressions
      i. [ (“1”, 2) ; (“3”, 4) ] Type =
      ii. fun f a -> [a ; a+1] Type =
   b. (3 pts each) Write an OCaml expression with the following type
      i. int * int list Code =
      ii. int list -> (int -> int) Code =
   c. (3 pts each) Give the value of the following OCaml expressions. If an error exists, describe the error.
      i. [1;2]::[3] Value =
      ii. let x y = y 3 in x (fun z -> z – 1) Value =
2. (18 pts) OCaml Programming

Solve the following OCaml programming problems. You are allowed to use List.rev (reverses a list) and the following (curried) map and fold functions, but no other OCaml library functions. Your solution must run in O(n) time for input lists of length n.

| let rec map f l = match l with       | let rec fold f a l = match l with       |
| [ ] -> [ ]                          | [ ] -> a                                |
| l (h::t) -> (f h)::(map f t)        | l (h::t) -> fold f (f a h) t             |
| ;;                                    | ;;                                      |

a. (9 pts) Write a function `makeLists` which when applied to a list `lst`, creates a new list for every element of `lst`, returning the results in a single list. You may use map or fold if you wish, but it is not required.

Example: `makeLists [1;2;4] = [[1];[2];[4]]`

b. (9 pts) Using either map or fold and an anonymous function, write a function `over20` which when applied to a list of ints `lst`, returns a list of all elements of `lst` that are 21 or over (preserving their relative order in `lst`).

Example: `over20 [33;18;21;19] = [33;21]`
3. (18 pts) Context Free Grammars
   Consider the following grammar: $S \rightarrow aA\ |\ A A$
   $A \rightarrow bS\ |\ ca$

   a. (8 pts) What is the set of strings accepted by this grammar?

   b. (10 pts) Provide a derivation of the string “ababca” for this grammar.