CS314: Principles of Programming Languages
Written Assignment 1

Nov. 1st, 2022

Name: Aryan Patel

NetID: adp182

- The written assignment has a total of **8** points. There is a total of 4 pages.
- For partial credit, show all of your work and clearly indicate your answers.
- You can either annotate your solution on this document or put your solution in another text document (e.g. MS Word) with clear marks to label the answer to each question.
- If you do not know how to type λ, just write lambda in English.
- Submit a PDF version of your solution to Canvas (e.g. using the printing function of Word.)

OCaml

1. Provide OCaml expressions (without type annotations) that have the following types:

   (a) ($\frac{1}{2}$ **point) int → int list → bool list**

   ```
   let divisible n l =
     List.map (fun x -> if (x mod n) = 0 then true else false) l;;
   ```

   (b) ($\frac{1}{2}$ **point) 'a → ('a → 'b) → 'b**

   ```
   let apply x f = f x;;
   ```

2. ($\frac{1}{2}$ **point)** Define a function f that when used in the following expression will not produce any type errors:

   ```
   fold_left f ([], 0) [5;4;3;2;1]
   ```

   The implementation of fold_left is given for reference, below.

   ```
   let rec fold_left f a l = match l with
   | [] -> a
   | h :: t -> fold_left f ( f a h ) t
   ```

   ```
   let f acc hd =
     match acc with
     | (lst, sum) -> (hd :: lst, sum + hd);;
   ```

OCaml Semantics

3. **(2 points)** Refer to the Micro-OCaml language (conditionals included) in Slide 41 in the lecture on Operational Semantics. Give the derivation for the following judgment:

   •;**let** x = 6 **in if** eq0 (**let** x = 3 **in** x + 2) **then** 3 **else** x ⇒ 6

   ```
   x:6; x -> 6
   ————————————————————
   x:3; x -> 3
   x:3; 2 -> 2
   3 + 2 is 5
   ————————————————————
   3 -> 3
   x:3; x + 2 -> 5          5 != 0
   ————————————————————
   eq0 x:3; x + 2 -> false          x:6; 6 -> 6
   ————————————————————
   let x = 6 in if eq0 (let x = 3 in x + 2) then 3 else x -> 6
   ```

Lambda Calculus

4. **(1 point)** Choose whether the following statements are true or false:

   (a) ($\frac{1}{2}$ point) $\lambda x.\lambda y.y\ x$ is α-equivalent to $\lambda f.\lambda n.n\ f$ — A. True / B. False

   (b) ($\frac{1}{2}$ point) $\lambda y.y\ x$ is α-equivalent to $\lambda x.x\ y$ — A. True / B. False

5. **(1 point)** Reduce the following λ expression to normal form. Show each reduction step and write whether it is an α-reduction or β-reduction. If already in normal form, write "normal form".

$$(\lambda x.x\ (\lambda x.y\ x))\ (\lambda z.z)$$

(λx.x (λx.y x)) (λz.z)
λz.z (λx.y x)
λx.y x

beta reduction

6. **(1 point)** Reduce the following λ expression to normal form. Show each reduction step and write whether it is an α-reduction or β-reduction. If already in normal form, write "normal form".

$$(\lambda x.\lambda y.x\ y\ z)\ (\lambda c.c)\ ((\lambda a.a)\ b)$$

(λx.λy.x y z) (λc.c) ((λa.a) b)
(λy.λc.c y z) ((λa.a) b)
(λy.λc.c y z) b
λc.c b z
b z

beta reduction

7. **(1 point)** Reduce the following λ expression to normal form. Show each reduction step and write whether it is an α-reduction or β-reduction. If already in normal form, write "normal form".

$$(\lambda x.(\lambda y.(x\ y)))\ y$$

(λx.(λy.(x y))) y
(λx.(λz.(x z))) y - alpha reduction
(λz.(y z)) - beta reduction

λz.y z

8. **($\frac{1}{2}$ point)** Which of the following lambda terms has the same semantics as this bit of OCaml code (choose exactly one):

$$\textbf{let}\ \text{func}\ x = (\textbf{fun}\ y \rightarrow y\ x)\ a\ b$$

A. $(\lambda y.y\ x)\ a\ b$

B. $(\lambda x.(\lambda y.y\ x)\ a\ b)$

C. $(\lambda x.(\lambda y.y\ x))\ a\ b$

D. $(x(\lambda y.y\ x))\ a\ b$

Extra page for solutions.