University of Massachusetts Lowell — Comp 3010: Organization of Programming Languages

Assignment 3

DUE: Tuesday, Oct 8, 2024, 11:59PM

Submission instructions: We are using GradeScope for PDF submission and the coding assignment. That is, you must submit your work as a PDF document. If you are not able to submit your assignment, please contact course staff. Some points to note:

- Your submission must be a PDF. We encourage you to use LaTeX (see below). If you produce your assignment using a different tool, please make sure your assignment is legible and follows these submission instructions.
- The first page of your assignment should state your name, UML ID, and a list of collaborators (if any), and *all other pages of your other pages should not contain any identifying information*, i.e., they should not contain your name.
- We will provide a LaTeX template you can use to write the answers to your assignment.

1 Large-Step Semantics (25 points)

Let us define the calculator language below that performs arithmetic and boolean operations. It has three different syntactic classes: (1) arithmetic expressions a (2) boolean expressions b, and (3) final values v. The following questions require you define small-step semantics. That is, first you need to define the configuration. Then, you need to write inference rules that show one configuration large-steps to another configuration. Recall that your large-step is a relation between an expression and a value.

$$\begin{array}{lll} n & \in & \mathbb{Z} \\ a & ::= & n \mid a_1 + a_2 \mid a_1 \times a_2 \\ b & ::= & \textbf{true} \mid \textbf{false} \mid a = a \mid a \neq a \\ & & \mid a \leq a \mid a > a \mid \neg b \mid b \&\& b \\ v & ::= & n \mid \textbf{truefalse} \end{array}$$

(a) Write large-step semantics for the syntactic class of arithmetic expressions generated by a. Assume the standard addition and multiplication operation between integers.

(b) Write large-step semantics for the syntactic class of boolean expressions generated by b. Assume the standard boolean operations between integers as well as between boolean expressions.

2 OCaml Program (25 points)

Write a large-step interpreter for the calculator language defined in the previous question. Your file should be named "hw3.ml". It should define the following types and functions. Follow the comments; fill in the holes and submit your hw3.ml in gradescope.

```
(* Type representing arithmetic expressions.
 * Fill in the holes
 *)
type aexp =
| Int of _ (* Constructor for n *)
| Add of _ (* Constructor for a_1 + a_2 *)
| Mul of _ (* Constructor for a_1 * a_2 *)

exception NoRuleApplies

(* Implement the following function by removing the exception.
*
* val lstep_aexp : aexp -> aexp
*
* It should take an arithmetic expression a and return a
   arithmetic expression a'
* such that a large steps to a value
*)
let lstep_aexp a = raise NoRuleApplies


(* Type representing boolean expressions.
* Fill in the holes
*)
type bexp =
| True (* Constructor for true *)
| False (* Constructor for false *)
| Eq of _ (* Constructor for a = a *)
| Neq of _ (* Constructor for a != a *)
| Leq of _ (* Constructor for a <= a *)
| Gt of _ (* Constructor for a > a *)
| Neg of _ (* Constructor for !a *)
| And of _ (* Constructor for a && a *)



(* Implement the following function by removing the exception
*
* val lstep_bexp : bexp -> bexp
*
* It should take a boolean expression b and return a boolean
   expression b'
* such that b large steps to a value
*)
let lstep_bexp b = raise NoRuleApplies
```