Homework: Lazy Evaluation
Optional Assignment B: Can increase your overall grade by up to 10%

Due-date: April 21 at 11:59pm
Submit online on Blackboard LS

Homework must be individual's original work. Collaborations and of any form with any students or other faculty members are not allowed. If you have any questions and/or concerns, post them on Piazza and/or ask our TA or me.

Learning Outcomes

- Knowledge and application of Functional Programming
- Ability to understand syntax specification
- Ability to design software following requirement specifications (operational semantics)

Question

Consider the grammar as specified in Homework assignment 5.

1. Write a function `eval` which takes as input a syntactically correct program as per the grammar, an environment, a heap and computes the semantics. Follow the semantics of dynamic scoping as discussed in class with the following modification. Evaluate expression if and only if the evaluation of the expression is necessary. For instance,

   (a) The result of a computation may not necessarily require the evaluation of all operands in the condition. As an example consider a condition expression of the form:

   ```
   ( (and (gt 1 2) (or (lt 1 2) (eq 1 1)))
     (+ 1 2)
     (+ 1 3)
   )
   ```

   The condition `(gt 1 2)` evaluates to false. As a result, the evaluation of the condition `(or (lt 1 2) (eq 1 1))` is not necessary.

   (b) The evaluation of the expressions appearing as arguments of an application are evaluated if and only if their mapping to actual argument is necessary for the computing the result of the application. As an example, consider the application of a function of the following form:

   ```
   (fun ((f (x y z)) ((gt z y) x y))
        (apply (f ((+ 2 2) (+ 4 4) (+ 0 1)))))
   ```

   As the condition `(gt z y)` evaluates to false (based on the evaluation of `z` and `y`), the expressions `(+ 2 2)` and `x` are never evaluated.

50% of the test cases that will used for assessing your submission will not use any reference operations.

2. If we assume that all variables used in any function are either declared locally within the function or appear as function parameters, does the above evaluation strategy (we call it *Lazy* evaluation) result in a different semantics (i.e., result of computation) compared to the ones we have done in the Homework assignments 4 or 5.

   Write your answer as Racket comment in the file `<netid>.rkt`. You can use multi-line comments enclosed in #| and |#.

Organization and Submission Guidelines You will have a file named `program.rkt`, which will contain the input programs.

You will write the functions necessary for this assignment in a file named `<netid>.rkt`. At the top of this file, you must include

```
#lang racket
(require "program.rkt")
(provide (all-defined-out))
```

Use the `program.rkt` and `<netid>.rkt` in the same folder; which will allow you to access the definitions of the input program easily.

You are required to submit the `<netid>.rkt` file only on the blackboard. If you netid is asterix, then the name of the file should be asterix.rkt. You must follow the instructions for submission. For instance, do not submit the file with any other extension, do not submit any other file, do not submit any zip files, remove/comment any inclusion of test code including trace directives that get automatically executed when your code is tested for evaluation, remove incomplete code.

In terms of the racket language features you can use, the same rules continue to apply for this assignment as well: list, numbers, boolean and typical operations over them, if-then-else and cond.

If you do not follow the requirements or submission guidelines of the assignment, then your submission may not graded. If you have doubts about the instruction, please post/ask.