Spring, 1992 Name: ______________________

CS 541 — Programming Languages 1

Operational Semantics Test (Version 1)

Before you begin, please take a moment to look over the entire test so that you can budget your time.
If you get an incomplete on this test, you can get at most a 'B' on the makeup test.
The usual directions apply.

1. Give a terminal transition system for the language described below. You should include a description of
   (a) (5 points) the set of configurations (Γ),
   (b) (5 points) the set of final configurations (T),
   (c) (15 points) the axiom schemes and inference rules that define the transition relation ($\Rightarrow$),
   (d) (5 points) the input ($\mathcal{I}$) function, and
   (e) (5 points) output ($\mathcal{O}$) function.

   You may give either "big step" or "little step" rules for $\Rightarrow$.

   The language's grammar and informal definition are as follows.

   $\langle expr \rangle$ ::= `loopForever` | `true` | `false`
   | `(not` $\langle expr \rangle$`)`
   | `(`$\langle expr \rangle$ `or` $\langle expr \rangle$`)`

   The meaning of `loopForever` is an infinite loop (nontermination). The meaning of `true`, `false`, and `not` are as usual. The `or` in this language is to be evaluated with a symmetric version of short-circuit evaluation; that is, the meaning of `(`e_1 `or` e_2`)` is:

   - *true*, if either e_1 or e_2 means *true*,
   - the meaning of e_2, if e_1 means *false*,
   - the meaning of e_1, if e_2 means *false*,
   - an infinite loop, if both e_1 and e_2 mean infinite loops or if either e_1 or e_2 is *false* and the other is an infinite loop.

   For example, `(not (not (not true)))` means *false*, `(not loopForever)` is an infinite loop, `(true or loopForever)` means *true*, `(loopForever or false)` is an infinite loop, and `((not true) or loopForever)` is an infinite loop.

   Check that your semantics does not allow looping on expressions that do not mean infinite loops.