Programming Languages

Qualifiers Examination
Spring 2008

December 12, 2007

Problem 1 [10 pts]

Using the sequent notation (also known as big-step) studied in class, provide the operational semantics for the following loop command

```
do command while expression
```

Problem 2 [70 pts]

Consider the following simplified syntax for a flow-chart language

```
<program>    ::= 1: start  <commands>
<commands>   ::= <label>: stop
             |   <label>: <command> ; <commands>
<command>    ::= <identifier> := <expression>
             |   goto <label>
             |   if <expression> goto <label>
<expression> ::= <number>
             |   <identifier>
             |   pred(<expression>)
             |   succ(<expression>)
```

Provide a complete denotational semantics for this language. Use the following guidelines

- use only one type (Nat) and there are no variable declarations

- assume that each program has its command labeled with distinct and consecutive labels (i.e., `start` has label 1, the next command has label 2, etc.) and it is syntactically correct with respect to the labels (e.g., there are no jumps to non-existant labels).

Problem 3 [30 pts]

Write a flow-chart program that computes the quotient and remainder of the division x/y using *only* predecessor and successor operations (you are allowed to use arbitrary comparisons in the tests), and under the assumption that $x \geq 1$ and $y \geq 1$.

Prove it to be partially correct using the inductive assertions method (from the Floyd approach) and the specification:

Precondition: $x \geq 1 \wedge y \geq 1$

Postcondition: $q = x/y \wedge m = x \; mod \; y$

(Remember that by x/y we intend the integer division between the two numbers).

Derive appropriate floydian expressions and show how to use them to prove total correctness.