Assignment 2: Program it Out
15-414/15-424 Bug Catching: Automated Program Verification

Due: **11:59pm**, Thursday 9/13/18
Total Points: 50

1. **Count the states (5 points)** How many intermediate and final states does the following program pass through while executing from initial state ω with $\omega(x) = 0$ and $\omega(y) = 1$:? Why?

$$x := x + 1; \texttt{while}(x < 100)\ \{x := 2 * x; y := x - 1\}; x := -x;$$

2. **Count the axioms (5 points)** Let α denote the program from Problem 1. If you were asked to write a sequent calculus proof of the following formula, how many times would you need to invoke the [unwind] axiom? Explain your answer to recieve credit.

$$\vdash x = 0 \land y = 1 \to [\alpha] y > 100$$

3. **Risky business (10 points)** Recall from lecture that we discussed why it's best to use the assignment axiom working from the innermost assignment first. Show how to complete the proof when working from the other direction, starting from the outermost assignment first. Be careful about variable capture!

$$\begin{array}{lr}
 & \dots \\ \hline
\text{[:=]} & x{=}a \land y{=}b \vdash [x := x + y][y := x - y][x := x - y](x = b \land y = a) \\ \hline
\text{[;]} & x{=}a \land y{=}b \vdash [x := x + y][y := x - y; x := x - y](x = b \land y = a) \\ \hline
\text{[;]} & x{=}a \land y{=}b \vdash [x := x + y; y := x - y; x := x - y](x = b \land y = a) \\ \hline
\to\text{R} & \vdash x = a \land y = b \to [x := x + y; y := x - y; x := x - y](x = b \land y = a)
\end{array}$$

4. **Missing condition (10 points)** Fill in the missing condition in the `if` statement to make the following dynamic logic formula valid. Then use the axioms of dynamic logic in sequent calculus to prove it.

$$[\texttt{if}(\ldots)\{r := r - w; q := q + 1\}\ \texttt{else}\ \{?\texttt{false}\}](wq + r = x \land 0 \leq r)$$

It is best to go about this problem by starting to prove the formula first, and using your partial proof to find the right condition.

5. **New feature (5 points)** When rummaging through the syntax manual of other imperative programming languages and comparing them to the language considered in class, a clever student found that we totally neglected her favorite feature of addition-assignment. The addition-assignment $x \mathrel{+{:}{=}} e$ adds e to the current value of x, and stores the result in x. Your job is to define a semantics $[\![x \mathrel{+{:}{=}} e]\!]$ for this construct as the set of all pairs of initial and final states of running $x \mathrel{+{:}{=}} e$.

6. **New axiom (5 points)** After now having added the conditional assignment $x \mathrel{+{:}{=}} e$ as a statement to the programming language, your next task is to design an axiom for it:

$$([\mathrel{+{:}{=}}])\quad [x \mathrel{+{:}{=}} e]p(x) \leftrightarrow \ldots$$

7. **Use it! (10 points)** Use your proof axiom $[\mathrel{+{:}{=}}]$ to conduct a sequent calculus proof for the formula:

$$x = y \to [x \mathrel{+{:}{=}} y; x \mathrel{+{:}{=}} -2 \cdot x + y]\, x = -y$$