Review Exercises (not to hand in—midterm prep.) **cs2me3 Winter 2004**

1. Recall Kruskal's algorithm. Define T to be promising iff $T \cup \{\text{edges not yet considered}\}$ is connected. More precisely, suppose that $e_1, e_2, \ldots, e_m$ are the edges sorted in non-decreasing order of weights. Then, at stage i of the algorithm, when edge i has been considered, T remains promising iff $T \cup \{e_{i+1}, e_{i+2}, \ldots, e_m\}$ is connected. Show that "T is promising" is a loop invariant of Kruskal's algorithm. Then show how this implies that at the end, (V, T) is a connected graph if the original input graph was connected. That is, show (informally):

$$\{G = (V, E) \text{ is connected}\}\texttt{Kruskal's Algorithm}\{(V, T) \text{ is connected}\}$$

2. Consider the following program for division:

```
int q=0;
int r=x;
while (y <= r) {
   r=r-y;
   q=q+1;
}
```

The above program, on input $x, y \geq 0$, computes $q = \text{div}(x, y)$ and $r = \text{rem}(x, y)$. What would be good pre- and post-conditions for this program? What would be a good loop invariant? Prove that your loop invariant holds, and that it implies the postcondition from the precondition.

3. Recall the definition of regular expressions given on assignment 2. Show that the set of strings given by the grammar $G = (\{S\}, \{a\}, S, \{S \longrightarrow aSa|a\})$ can be given by a regular expression.

4. Show that CFG and BNF grammars are equivalent. Give a BNF grammar for the set of strings of 0s and 1s which contain at least three consecutive 1s.

5. Verify each of the following. For the "$\not\models$" cases, give specific formulas, models, and object assignments for which the logical consequence fails. For the "$\models$" cases, give a proof based on the Basic Semantic Definitions. Where there is no $\models$, show that the formula is valid.

   (a) $\neg\forall x\alpha \leftrightarrow \exists x\neg\alpha$

   (b) $\neg\exists x\alpha \leftrightarrow \forall x\neg\alpha$

   (c) $(\forall x\alpha \wedge \forall x\beta) \leftrightarrow \forall x(\alpha \wedge \beta)$

   (d) $\exists x(\alpha \vee \beta) \leftrightarrow (\exists x\alpha \vee \exists x\beta)$

   (e) $\exists x(\alpha \wedge \beta) \models (\exists x\alpha \wedge \exists x\beta)$

   (f) $(\exists x\alpha \wedge \exists x\beta) \not\models \exists x(\alpha \wedge \beta)$

(g) $\forall x \forall y \alpha \leftrightarrow \forall y \forall x \alpha$

(h) $\exists x \exists y \alpha \leftrightarrow \exists y \exists x \alpha$

(i) $\exists y \forall x \alpha \vDash \forall x \exists y \alpha$

(j) $\forall x \exists y \alpha \nvDash \exists y \forall x \alpha$

(k) $\forall x \alpha \vDash \exists x \alpha$ (what specific requirement on the universe of discourse makes it true?)

(l) $\forall x \forall y (x = y \supset fx = fy)$ for any unary f

(m) show that $\forall x \forall y (fx = fy \supset x = y)$ is **not** valid

6. Show that $\{P0, Ps0, Pss0, Psss0, Pssss0, \ldots\} \nvDash \forall x Px$. Here $\mathcal{L} = [0, s; P]$, where 0 is a 0-ary function, s is a unary function, and P is a unary predicate.

7. Show that $\forall x(gfx = x)$ is **not** a logical consequence of $\forall x(fgx = x)$. Here $\mathcal{L}$ is *some* first order language with two unary functions: f, g.

8. Let $\mathcal{M}$ be a structure and let Φ be the set of all sentences α such that $\mathcal{M} \vDash \alpha$ (i.e., Φ is the set of all those sentences that are satisfied by the structure $\mathcal{M}$). Show that if $\Phi \vDash \beta$, then $\beta \in \Phi$.

9. Give a sentence α over the language $\mathcal{L} = [;=]$ (i.e., the language $\mathcal{L}$ has only one predicate symbol, "$=$"), such that $\mathcal{M} \vDash \alpha$ iff M has exactly three elements.

10. Give a satisfiable sentence α in the language $\mathcal{L} = [;R]$, where R is a binary predicate, such that α has no finite model. (**Hint.** Think of R as something like "$<$".)

11. Consider any first order language $\mathcal{L} = [f_1, f_2, \ldots, f_n; P_1, P_2, \ldots, P_m]$ where the arity of the function symbol f_i is k_i, and the arity of the predicate P_i is l_i. Show that the set of $\mathcal{L}$-formulas can be generated by a grammar of type 2 (a Context-Free Grammar). Remember that there is one problem: we have an infinite set of variables at our disposal when constructing $\mathcal{L}$-formulas (by structural induction), but a grammar has finitely many terminals; how can we resolve this problem?