# CS 310, Assignment 5

Answers

1. Consider the following context-free grammar with start symbol $S$, nonterminals $\{S, X, C, B\}$, and terminals $\{a, b\}$:

$$S \rightarrow Xb \qquad X \rightarrow C \qquad X \rightarrow Ba \qquad C \rightarrow a \qquad C \rightarrow \varepsilon \qquad B \rightarrow b$$

   (a) Compute *all* the sets FIRST and FOLLOW necessary to implement a recursive decent parser for this grammar.

   ---

   ANSWER: We do not need to compute any sets for $S$ (there is a single rule for that symbol).
   $X$ has two rules and it is also the case that $X \Rightarrow \varepsilon$ so we need:

$$\text{FIRST}(C) = \{a, EOS\} \qquad \text{FIRST}(Ba) = \{b\} \qquad \text{FOLLOW}(X) = \{b\}$$

   $C$ also has two rules, one of them an $\varepsilon$-rule, so we need:

$$\text{FIRST}(a) = \{a\} \qquad \text{FOLLOW}(C) = \{b\}$$

   ---

   (b) Is this grammar suitable for recursive descent parsing? Justify your answers formally based on the sets computed in the previous question.

   ---

   ANSWER: The grammar is not suitable for recursive descent parsing since $\text{FIRST}(Ba) \cap \text{FOLLOW}(X) = \{b\} \neq \emptyset$.

   ---

2. What should the pre-condition $P$ be in each of the following correctness statements for the statement to be an instance of Hoare's assignment axiom scheme? All variables are of type `int`.

   (a) $P$ `{ x = 1; } x <= 2`

   (b) $P$ `{ x = 2; } x <= 1`

   (c) $P$ `{ x = x + y; } x*x > 5`

   (d) $P$ `{ x = x + y; } ForAll (y=0; y<10) x*y > 0`

   (e) $P$ `{ x = x + y; } Exists (x=0; x<100) x*y >= x+y-z`

(f) $P$ `{ x = x + y; } Exists (y=0; y<x) z+y >= x`

---

ANSWER:

(a) `1 <= 2` or `true`

(b) `2 <= 1` or `false`

(c) `(x+y)*(x+y) > 5` (note the parentheses)

(d) `ForAll (k=0; k<10) (x+y)*k > 0` (we need to rename the bound variable `y` because its name clashes with the `y` introduced by the assignment)

(e) `Exists (x=0; x<100) x*y >= x+y-z` (no change, since all the occurrences of `x` refer to the bound variable; this can be verified by renaming the bound variable `x`)

(f) `Exists (k=0; k<x) z+k >= x+y` (the bound variable `y` was renamed again because a free variable with the same name is introduced by the substitution)

---

3. Consider the following correctness statements where $S$ is any terminating code. In each case give the least restrictive conditions that the code $S$ has to satisfy in order to make the correctness statement valid.

(a)
```
ASSERT( false )
S
ASSERT( true )
```

---

ANSWER: The statement is valid for any terminating code $S$ since the post-condition holds no matter what.

---

(b)
```
ASSERT( false )
S
ASSERT( false )
```

---

ANSWER: The statement is valid for any terminating code $S$.

Indeed, the statement says that when the code is started in a state satisfying assertion `false`, then when the code terminates it will be in a state satisfying assertion `false`. Since no state can satisfy the pre-condition, the implication holds vacuously.

---