In Class Exercises – Axiomatic Semantics

1. Classify the following Hoare triples as valid or invalid, and explain why briefly. Examples:

- \( \{ \text{true} \} \ x := x \ast x \ \{ x \geq 0 \} \) VALID; \( x^2 \) is always \( \geq 0 \).
- \( \{ x \geq 0 \} \ x := x - 1 \ \{ x \geq 0 \} \) INVALID: If \( x = 0 \) at the start of the program, then \( x \geq 0 \) but \( x - 1 < 0 \).

(a) \( \{ \text{true} \} \ x := 3 \ \{ x = 4 \} \)
(b) \( \{ \text{true} \} \ x := 3 \ \{ \text{false} \} \)
(c) \( \{ \text{false} \} \ x := 3 \ \{ \text{false} \} \)
(d) \( \{ x > y \land x > 0 \} \ y := x + y \ \{ y > 0 \} \)
(e) \( \{ y > x \land x > 0 \} \ x := x + y \ \{ x > y \} \)
(f) \( \{ x = 3 \} \text{ while odd}(x) \text{ do skip} \ \{ \text{true} \} \)
(g) \( \{ x \text{ is even} \} \text{ while odd}(x) \text{ do skip} \ \{ \text{true} \} \)
(h) \( \{ \text{false} \} \text{ while odd}(x) \text{ do skip} \ \{ x \text{ is even} \} \)

2. Use the assignment rule to find a precondition \( Q \) that makes each triple valid.

(a) \( \{ Q \} \ x := x \ast 3 \ \{ x = 15 \} \)
(b) \( \{ Q \} \ x := 14 \ \{ x = 14 \} \)
(c) \( \{ Q \} \ r := r - 1 \ \{ r^3 + \sin(r) - 12 = 4 \} \)
(d) \( \{ Q \} \ y := y + 1; x := x + a[y] \ \{ x = \sum_{k=1}^{y} a[k] \} \)

3. Fill in the blanks with the appropriate assertions to come up with \( Q \). Use the pairs of blank lines to show uses of the consequence rule (i.e., to show simplifications).

\[
Q:\quad \underline{\quad}\quad \\
\text{if } w = z \text{ then begin} \\
\quad \underline{\quad}\quad \underline{\quad}\quad \\
w := 4; \\
\quad \underline{\quad}\quad \underline{\quad}\quad \\
z := z + 2 \\
\text{end else begin} \\
\quad \underline{\quad}\quad \underline{\quad}\quad \\
z := w \ast z; \\
\quad \underline{\quad}\quad \underline{\quad}\quad \\
w := 2 \ast w \\
\text{end} \\
\{ z = 5 \ast w \}
4. To prove the loop

{\text{pre } Q} \\
\text{Initialization} \\
{\text{inv } P} \\
\text{while } B \text{ do } S; \\
{\text{post } R}

correct, we use the following steps:

- Prove \{Q\} Initialization \{P\} is valid
- Prove P is invariant: \{P \land B\} S \{P\}
- Prove that if the loop stops, R holds: \(P \land \neg B \Rightarrow R\)
- Prove bound function \(t\) decreases with each iteration.
- Prove that if there are iterations left then \(t > 0\): \(P \land B \Rightarrow t > 0\)

Prove the following loop correct:

\{0 < n\} \\
i := 1; \\
{\text{inv } P}: \ (0 < i \leq n) \land (i \text{ is a power of 2}) \\
{\text{bound function}}: \ n - i \\
\text{while } 2 \times i \leq n \text{ do } i := i \times 2; \\
\{(0 < i \leq n < 2 \times i) \land (i \text{ is a power of 2})\}

5. We can also go the other way: Given a precondition, invariant, and postcondition, we can develop a corresponding loop using the following procedure:

- Find a loop guard \(B\). (“When are we done?”)
- Find initialization to establish invariant \(P\).
- Guess bound function \(t\) and find ways to decrease it. (Make sure it meets the requirements listed in the first section.)
- Ensure that \(P\) is reestablished.

Write a program that, given \(n\), calculates

\[
F_n = \begin{cases} 
0 & \text{if } n = 0 \\
1 & \text{if } n = 1 \\
F_{n-1} + F_{n-2} & \text{if } n > 1
\end{cases}
\]

Given the following specification:

\{\text{pre } Q: \ n > 0\} \\
\{\text{inv } P: \ 1 \leq i \leq n \land a = F_i \land b = F_{i-1}\} \\
\{\text{post } R: \ a = F_n\}