CS615 Practice Assignment 2

- ***This assignment is intended to help you get a better understanding of techniques learnt in class by applying them to actual problems.***
- ***Do not turn in solutions to this assignment. These will not be graded.***
- ***Mutual discussion among students is strongly encouraged. However, it is recommended that you finally solve all the problems yourself. Mutual discussion should only be for purposes of clarifying doubts and concepts.***
- ***You are welcome to approach the TA and the instructor for clarifying doubts. Please send an email asking for an appointment.***

1. The following program, C, is intended to compute the root of a simple quadratic equation. All variables in the program are of type `natural_number`.

```
# command C
x := 0;                              # command C0
res := (a - x) * (a - x);            # command C10   #  |
                                                     #  |
while (res <> 0)                     # command C11   # command C1
  res := res - 2 * (a - x) + 1;      # command C110 #  |
  x := x + 1;                        # command C111 #  |
```

We wish to prove the correctness of the following Hoare Logic formula: $\{P\}\ C\ \{Q\}$ where $P = (a > 0)$ and $Q = (x = a)$.

Give a proof for the above formula using the Hoare deductive system. State clearly any domain-specific axioms that you need in order to complete the proof. You must minimize the use of such axioms.

Solution:

- The proof goal $\{P\}C\{Q\}$ must first be decomposed into three proof subgoals using the *consequence rule* of Hoare Logic: $P \Rightarrow P_1$, $\{P_1\}C\{Q_1\}$ and $Q_1 \Rightarrow Q$.
- Since $C = C0; C1$, the proof goal $\{P_1\}C\{Q_1\}$ must then be decomposed into $\{P_1\}C0\{P_2\}$ and $\{P_2\}C1\{Q_1\}$ using the *composition rule* of Hoare Logic.
- Using the *backward assignment axiom* for $\{P_1\}C0\{P_2\}$, we find that $P_1 = P_2[x \leftarrow 0]$. Clearly, P_2 contains the clause $(x = 0)$. Thus, P_1 must contain the clause $(0 = 0)$.
  - Since we must also have $P \Rightarrow P_1$, and $P = (a > 0)$, therefore P_1 must contain the clauses $(a > 0)\ \wedge\ (0 = 0)$.
  - Also, since $P_1 = P_2[x \leftarrow 0]$, this implies P_2 contains the clauses $(a > 0)\ \wedge\ (x = 0)$.
- Using the *composition rule*, $\{P_2\}C1\{Q_1\}$ must next be decomposed into $\{P_2\}C10\{P_3\}$ and $\{P_3\}C11\{Q_1\}$.
- Using the *backward assignment axiom* for $\{P_2\}C10\{P_3\}$, we find that $P_2 = P_3[res \leftarrow (a-x)^2]$. Since P_3 contains the clause $(res = (a-x)^2)$, we find that P_2 must contain the clause $((a-x)^2 = (a-x)^2)$.

- Using the information about clauses in P_2 already obtained earlier, we find that P_2 contains the clauses $(a > 0) \wedge (x = 0) \wedge ((a - x)^2 = (a - x)^2)$.
- Since $P_2 = P_3[res \leftarrow (a - x)^2]$, it now follows that P_3 must contain the clauses $(a > 0) \wedge (x = 0) \wedge (res = (a - x)^2)$
- Since $P_1 = P_2[x \leftarrow 0]$, therefore P_1 must contain the clauses $(a > 0) \wedge (0 = 0) \wedge (a^2 = a^2)$. Note that we have used an axiom from natural number arithmetic here, namely: $(a - 0) = a$.

* We must now focus on $\{P_3\}C11\{Q_1\}$. Since $C11$ is a while loop, so using the *while rule* and *consequence rule*, this proof goal is decomposed into $P_3 \Rightarrow Linv$, $\{Linv \wedge (res <> 0)\}C110; C111\{Linv\}$, and $Linv \wedge (res = 0) \Rightarrow Q_1$.

  Recall also that $Q_1 \Rightarrow Q$ must hold. If we choose $Q_1 = Q = (x = a)$, then we simply need to obtain an $Linv$ satisfying the three constraints described above in order to complete the proof.

  This is where domain knowledge and axioms from the domain can come in useful to help us choose the right $Linv$. Of course, any $Linv$ (even if accidentally chosen and not from domain knowledge) that satisfies the above constraints will suffice to complete the proof. Note also that the above system of three constraints serves as a guideline when choosing an $Linv$. An assertion that violates these three constraints cannot be used as $Linv$ in order to complete the proof. Similarly, if one can show that regardless of the assertion $Linv$, the above system of three constraints in incosistent (cannot be simultaneously satisfied), it automatically implies that the correctness of $\{P\}C\{Q\}$ cannot be deduced using the deductions we have made thus far.

  - A specific choice of $Linv$ in this case is $(res = (a - x)^2) \wedge (x \leq a)$.
    I will leave it to you now to show that with this choice of $Linv$, all the following three constraints are satisfied.
    (a) $P_3 \Rightarrow Linv$
    (b) $\{Linv \wedge (res <> 0)\}C110; C111\{Linv\}$
    (c) $Linv \wedge (res = 0) \Rightarrow Q_1 = Q = (x = a)$

  Note that in proving these constraints, you will need to use certain axioms from arithmetic on natural numbers. You should identify the minimal set of such axioms needed.

* This completes the proof of $\{P\}C\{Q\}$.

Can you think of another choice of $Linv$ that will also complete the proof above? What if you used $Linv = (res \leq (a - x)^2) \wedge (x \leq a)$? Can you think of other candidates?

2. Consider the program given below, where all variables are of type `natural_number`:

```
# command C
temp := m;
temp1 := m;

while (temp1 > 0)
   temp := temp + m;
   temp1 := temp;

   while (temp1 > n)
      temp1 := temp1 - n;
```

We wish to prove the truth of the Hoare Logic formula $\{P\}C\{Q\}$ where $P = (m > 0 \wedge n > 0 \wedge n > m)$ and $Q = (temp = lcm(m, n))$. Here, lcm denotes the least common multiple of two non-zero natural numbers.

What is the minimum set of domain specific axioms that are needed (in addition to the *skip axiom*, *backward assignment axiom* and *random assignment axiom* of the Hoare deductive system) in order to complete the above proof?

You must also show the proof using the Hoare deductive system and using only the axioms that you have chosen above.