Ameya Acharya (apa52) & Quinn Beightol (qeb2)
CS 4110
Homework 4 Due Wednesday, October 1, 2014

1. (a)

$$\begin{aligned}
\mathcal{C}[\![\texttt{if } b \texttt{ then } c]\!] &= \{(\sigma,\sigma) \mid (\sigma,\texttt{false}) \in \mathcal{B}[\![b]\!]\} \cup \{(\sigma,\sigma') \mid (\sigma,\texttt{true}) \in \mathcal{B}[\![b]\!] \wedge (\sigma,\sigma') \in \mathcal{C}[\![c]\!]\} \\
\mathcal{C}[\![\texttt{do } c \texttt{ until } b]\!] &= \text{fix}(G) \\
\text{where } G(f) &= \{(\sigma,\sigma') \mid (\sigma,\sigma') \in \mathcal{C}[\![c]\!] \wedge (\sigma',\texttt{true}) \in \mathcal{B}[\![b]\!]\} \\
&\cup \{(\sigma,\sigma'') \mid (\sigma,\sigma') \in \mathcal{C}[\![c]\!] \wedge (\sigma',\texttt{false}) \in \mathcal{B}[\![b]\!] \wedge \exists\sigma''.(\sigma',\sigma'') \in f\}
\end{aligned}$$

(b)

$$\frac{\vdash \{P \wedge b\}\ c\ \{Q\} \qquad \vdash \{P \wedge \neg b\} \Rightarrow Q}{\vdash \{P\} \text{ if } b \text{ then } c\ \{Q\}} \text{One-armed If}$$

$$\frac{\vdash \{P\}\ c\ \{Q\} \qquad \vdash \{Q \wedge \neg b\}\ c\ \{Q\}}{\vdash \{P\}\ \texttt{do}\ c\ \texttt{until}\ b\{Q \wedge^{\neg} b\}} \text{Do}$$

2. (a) To show prove this equivalence, we will show that both expressions denote the same mathematical object. To make this easier to read, we will box the sections that we denote in later lines.

Denotation of (x := x + 21; x := x + 21):

$$\begin{aligned}
\mathcal{C}[\![x := x+21; x := x+21]\!] &= \{(\sigma,\sigma') \mid \exists\sigma''.((\sigma,\sigma'') \in \boxed{\mathcal{C}[\![x := x+21]\!]} \wedge (\sigma'',\sigma') \in \mathcal{C}[\![x := x+21]\!])\} \\
\mathcal{C}[\![x := x+21]\!] &= \{(\sigma,\sigma[x \mapsto n]) \mid (\sigma,n) \in \boxed{\mathcal{A}[\![x+21]\!]}\} \\
\mathcal{A}[\![x+21]\!] &= \{(\sigma,n) \mid (\sigma,n_1) \in \boxed{\mathcal{A}[\![x]\!]} \wedge (\sigma,n_2) \in \boxed{\mathcal{A}[\![21]\!]} \wedge n = n_1 + n_2\} \\
\mathcal{A}[\![x]\!] &= \{(\sigma,\sigma(x))\} \\
\mathcal{A}[\![21]\!] &= \{(\sigma,21)\} \\
\text{So: } \mathcal{A}[\![x+21]\!] &= \{(\sigma,n) \mid (\sigma,n_1) \in (\sigma,\sigma(x)) \wedge (\sigma,n_2) \in (\sigma,21) \wedge n = n_1 + n_2\} \\
&= \{(\sigma,n) \mid n = \sigma(x) + 21\} \\
&= \{(\sigma,\sigma(x)+21)\} \\
\text{And now: } \mathcal{C}[\![x := x+21]\!] &= \{(\sigma,\sigma[x \mapsto \sigma(x)+21])\}
\end{aligned}$$

Now that we have found the denotation of the first $\mathcal{C}[\![x := x+21]\!]$, we may use that in our $\mathcal{C}[\![x := x+21; x := x+21]\!]$:

$$\begin{aligned}
\mathcal{C}[\![x := x+21; x := x+21]\!] &= \{(\sigma,\sigma') \mid \exists\sigma''.((\sigma,\sigma'') \in \{(\sigma,\sigma[x \mapsto \sigma(x)+21])\} \wedge (\sigma'',\sigma') \in \\
&\quad \mathcal{C}[\![x := x+21]\!])\} \\
\mathcal{C}[\![x := x+21; x := x+21]\!] &= \{(\sigma,\sigma') \mid \exists\sigma''.\sigma'' = \sigma[x \mapsto \sigma(x)+21]\} \wedge (\sigma'',\sigma') \in \boxed{\mathcal{C}[\![x := x+21]\!]}\} \\
\mathcal{C}[\![x := x+21]\!] &= \{(\sigma'',\sigma''[x \mapsto n]) \mid (\sigma'',n) \in \boxed{\mathcal{A}[\![x+21]\!]}\} \\
\mathcal{A}[\![x+21]\!] &= \{(\sigma'',n) \mid (\sigma'',n_1) \in \boxed{\mathcal{A}[\![x]\!]} \wedge (\sigma'',n_2) \in \boxed{\mathcal{A}[\![21]\!]} \wedge n = n_1 + n_2\} \\
\mathcal{A}[\![x]\!] &= \{(\sigma'',\sigma''(x))\} \\
\mathcal{A}[\![x]\!] &= \{(\sigma'',\sigma(x)+21)\} \\
\mathcal{A}[\![21]\!] &= \{(\sigma'',21)\} \\
\text{So: } \mathcal{A}[\![x+21]\!] &= \{(\sigma'',n) \mid (\sigma'',n_1) \in (\sigma'',\sigma''(x)) \wedge (\sigma'',n_2) \in (\sigma'',21) \wedge n = n_1 + n_2\} \\
&= \{(\sigma'',n) \mid n = \sigma(x) + \sigma(x) + 21\} \\
&= \{(\sigma'',\sigma(x)+42)\} \\
\text{Now, we know: } \mathcal{C}[\![x := x+21]\!] &= \{(\sigma'',\sigma''[x \mapsto n]) \mid (\sigma'',n) \in \{(\sigma'',\sigma(x)+42)\}\} \\
\text{So, we conclude:} & \\
\mathcal{C}[\![x := x+21; x := x+21]\!] &= \{(\sigma,\sigma[x \mapsto \sigma(x)+42])\}
\end{aligned}$$

Denotation of (x := x + 42):

$$\begin{aligned}
\mathcal{C}[\![x := x+42]\!] &= \{(\sigma,\sigma[x \mapsto n]) \mid (\sigma,n) \in \boxed{\mathcal{A}[\![x+42]\!]\}} \\
\mathcal{A}[\![x+42]\!] &= \{(\sigma,n) \mid (\sigma,n_1) \in \boxed{\mathcal{A}[\![x]\!]} \wedge (\sigma,n_2) \in \boxed{\mathcal{A}[\![42]\!]} \wedge n = n_1 + n_2\} \\
\mathcal{A}[\![x]\!] &= \{(\sigma,\sigma(x))\} \\
\mathcal{A}[\![42]\!] &= \{(\sigma,42)\} \\
\text{So: } \mathcal{A}[\![x+42]\!] &= \{(\sigma,n) \mid (\sigma,n_1) \in (\sigma,\sigma(x)) \wedge (\sigma,n_2) \in (\sigma,42) \wedge n = n_1 + n_2\} \\
&= \{(\sigma,n) \mid n = \sigma(x) + 42\} \\
&= \{(\sigma,\sigma(x)+42)\} \\
\mathcal{C}[\![x := x+42]\!] &= \{(\sigma,\sigma[x \mapsto n]) \mid n = \sigma(x) + 42\} \\
&= \{(\sigma,\sigma[x \mapsto \sigma(x)+42])\}
\end{aligned}$$

The two are the same.

(b) To conserve space, we'll refer to (x:= 1; do x:= $x+1$ until $x<0$) as c_a, and we'll refer to while true do c as c_b.

$$\begin{aligned}
\mathcal{C}[\![c_a]\!] &= \{(\sigma,\sigma'') \mid \exists\sigma'.(\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:= }1]\!] \\
&\quad \wedge(\sigma',\sigma'') \in \mathcal{C}[\![\texttt{do }x\texttt{:= }x+1\texttt{ until }x<0]\!]\} \\
\mathcal{C}[\![x\texttt{:= }x+1]\!] &= \{(\sigma,n) \mid (\sigma,\sigma[x\mapsto n]) \in \mathcal{A}[\![x]\!]\} \\
\mathcal{A}[\![1]\!] &= \{(\sigma,1)\} \\
\mathcal{C}[\![x\texttt{:= }x+1]\!] &= \{(\sigma,\sigma[x\mapsto 1])\}
\end{aligned}$$

$$\begin{aligned}
\mathcal{C}[\![\texttt{do }x\texttt{:= }x+1\texttt{ until }x<0]\!] &= \text{fix}(G') \\
\text{where } G'(f) &= \{(\sigma,\sigma') \mid (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:= }x+1]\!] \wedge (\sigma',\texttt{true}) \in \mathcal{B}[\![x<0]\!]\} \\
&\cup\ \{(\sigma,\sigma'') \mid (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:= }x+1]\!] \wedge (\sigma',\texttt{false}) \in \mathcal{B}[\![x<0]\!] \\
&\quad \wedge\exists\sigma''.(\sigma',\sigma'') \in f\}
\end{aligned}$$

Going forward, we'll argue that the second set in the definition of G' (more precisely, the set defined after the first union operator) never adds anything to f_n–i.e. we'll prove $\forall n \in \mathbb{N}.P(n) = \{(\sigma,\sigma'') \mid ... \wedge \exists\sigma''.(\sigma',\sigma'') \in f_n\} = \emptyset$ using induction:

P(0): In this case $f_n = \emptyset$ so its impossible to have a pair (σ',σ'') satisfying (σ',σ''), so the the second set is the empty set.

Inductive Step (i.e. $P(n-1) \to P(n)$): Using the inductive hypothesis, we conclude that

$$\begin{aligned}
f_n = &= \{(\sigma,\sigma') \mid (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:= }x+1]\!] \wedge (\sigma',\texttt{true}) \in \mathcal{B}[\![x<0]\!]\} \\
&\cup\ \{(\sigma,\sigma'') \mid (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:= }x+1]\!] \wedge (\sigma',\texttt{false}) \in \mathcal{B}[\![x<0]\!] \\
&\quad \wedge\exists\sigma''.(\sigma',\sigma'') \in f_{n-1}\} \\
&= \{(\sigma,\sigma') \mid (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:= }x+1]\!] \wedge (\sigma',\texttt{true}) \in \mathcal{B}[\![x<0]\!]\} \\
&\cup\ \{(\sigma,\sigma'') \mid (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:= }x+1]\!] \wedge (\sigma',\texttt{false}) \in \mathcal{B}[\![x<0]\!] \\
&\quad \wedge\exists\sigma''.(\sigma',\sigma'') \in \emptyset\}
\end{aligned}$$

$$f_n = \{(\sigma,\sigma') \mid (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:= }x+1]\!] \wedge (\sigma',\texttt{true}) \in \mathcal{B}[\![x<0]\!]\}$$

And now we can re-write the second set in $G'(f)$:

$$\begin{array}{rcl}\{(\sigma,\sigma'') & \mid & (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:=}\ x+1]\!] \\ & \wedge & (\sigma',\texttt{false}) \in \mathcal{B}[\![x<0]\!] \\ & \wedge & \exists\sigma''.(\sigma',\sigma'') \in f_n\} \\ \\ =\{(\sigma,\sigma'') & \mid & (\sigma,\sigma') \in \mathcal{C}[\![x\texttt{:=}\ x+1]\!] \\ & \wedge & (\sigma',\texttt{false}) \in \mathcal{B}[\![x<0]\!] \\ & \wedge & \exists\sigma''.((\sigma',\sigma'') \in \mathcal{C}[\![x\texttt{:=}\ x+1]\!] \\ & \wedge & (\sigma'',\texttt{true}) \in \mathcal{B}[\![x<0]\!])\}\end{array}$$

Notice that the last three requirements on the stores are contradictory. Consider some σ' such that $\mathcal{B}[\![x<0]\!]\sigma' = \texttt{false}$. Looking at at the definition of $\mathcal{B}[\![x<0]\!]$ we can infer a few facts about σ':

$$\begin{array}{rcl}\mathcal{B}[\![x<0]\!] & = & ...\cup\{(\sigma',\texttt{false}) \mid (\sigma',n_1)\in\mathcal{A}[\![x]\!] \wedge (\sigma',n_2)\in\mathcal{A}[\![0]\!] \wedge n_1 \geq n_2\} \\ \mathcal{A}[\![x]\!] & = & \{(\sigma,\sigma(x))\} \\ \mathcal{A}[\![0]\!] & = & \{(\sigma,0)\}\end{array}$$

Since $\mathcal{B}[\![x<0]\!]\sigma' = \texttt{true}$, we know that $n_1 = \sigma'(x)$, $n_2 = 0$, and $\sigma'(x) \geq 0$. Now let's take a look at what happens when execute $x\texttt{:=}\ x+1$ on the store and then evaluate $x<0$:

$$\begin{array}{rcl}\mathcal{C}[\![x\texttt{:=}\ x+1]\!] & = & \{(\sigma,\sigma[x\mapsto n]) \mid (\sigma,n)\in\mathcal{A}[\![x+1]\!]\} \\ \mathcal{A}[\![x+1]\!] & = & \{(\sigma,n') \mid (\sigma,n_1')\in\mathcal{A}[\![x]\!] \wedge (\sigma,n_2')\in\mathcal{A}[\![x]\!] \wedge n' = n_1'+n_2'\}\end{array}$$

$$\begin{array}{rcl}\mathcal{A}[\![x]\!] & = & \{(\sigma,\sigma(x))\} \\ \mathcal{A}[\![1]\!] & = & \{(\sigma,1)\} \\ \mathcal{A}[\![x+1]\!] & = & \{(\sigma,\sigma(x)+1)\} \\ \mathcal{C}[\![x\texttt{:=}\ x+1]\!] & = & \{(\sigma,\sigma[x\mapsto\sigma(x)+1])\} \\ \mathcal{B}[\![x<0]\!] & = & \{(\sigma,\texttt{true}) \mid (\sigma,n_1)\in\mathcal{B}[\![x]\!] \wedge (\sigma,n_2)\in\mathcal{B}[\![0]\!] \wedge n_1 < n_2\} \cup \\ & & \{(\sigma,\texttt{false}) \mid (\sigma,n_1)\in\mathcal{B}[\![x]\!] \wedge (\sigma,n_2)\in\mathcal{B}[\![0]\!] \wedge n_1 \geq n_2\}\end{array}$$

which can be more compactly written:

$$\mathcal{B}[\![x<0]\!] = \begin{cases}\texttt{true} & \text{iff } \sigma'(x) < 0 \\ \texttt{false} & \text{otherwise}\end{cases}$$

Now the source of the contradiction should become clear. Let's evaluate our command on σ' and then evaluate our boolean expression:

$$\sigma'' = \mathcal{C}[\![x \texttt{:=}\ x+1]\!]\sigma' \quad = \quad \sigma'[x \mapsto \sigma'(x)+1])\}\mathcal{B}[\![x<0]\!]\sigma'' = \texttt{false}$$

The boolean expression evaluates to false because $\sigma'(x) > 0$ and so $\sigma''(x) = \sigma'(x)+1 > 0$. This, in turn, demonstrates that the three conditionals in the definition of the second set are contradictory. That set will be the empty set for all n. ✓
Now we can calculate the fixed point of G':

$$\begin{aligned}
\text{fix}(G') &= \bigcup_{i\geq 0}(G')^i(\emptyset) \\
&= \emptyset \cup \{(\sigma,\sigma') \mid (\sigma,\sigma') \in \mathcal{C}[\![x \texttt{:=}\ x+1]\!] \wedge (\sigma', \texttt{true}) \in \mathcal{B}[\![x<0]\!]\} \cup \emptyset \cup \\
&\quad\ \{(\sigma,\sigma') \mid (\sigma,\sigma') \in \mathcal{C}[\![x \texttt{:=}\ x+1]\!] \wedge (\sigma', \texttt{true}) \in \mathcal{B}[\![x<0]\!]\} \cup \emptyset \cup ... \\
&= \{(\sigma,\sigma') \mid (\sigma,\sigma') \in \mathcal{C}[\![x \texttt{:=}\ x+1]\!] \wedge (\sigma', \texttt{true}) \in \mathcal{B}[\![x<0]\!]\}
\end{aligned}$$

If we evaluate $\mathcal{C}[\![x \texttt{:= 1}]\!]$ on a store sigma, we'll get a new store $\sigma' = \sigma[x \mapsto 1]$, and then if we attempt to apply $\mathcal{C}[\![\texttt{do}\ x \texttt{:=}\ x+1\ \texttt{until}\ x<0]\!] = \text{fix}(G')$ to that store, well, we won't get a value–the relationship is undefined since $\sigma'(x) > 0$, which implies that $\sigma''(x) = \sigma'(x)+1 > 0$ and therefore $\mathcal{B}[\![x<0]\!]\sigma'' = \texttt{false}$. In other words, c_a doesn't terminate. Let's look at c_b and prove that it doesn't terminate as well (which in turn demonstrates that the commands are equivalent:

$$\begin{aligned}
\mathcal{C}[\![c_b]\!] &= \text{fix}(F) \\
F(f) &= \{(\sigma,\sigma) \mid (\sigma, \texttt{false}) \in \mathcal{B}[\![b]\!]\} \\
&\cup \ \{(\sigma,\sigma'') \mid (\sigma, \texttt{true}) \in \mathcal{B}[\![b]\!] \wedge (\sigma,\sigma') \in \mathcal{C}[\![c]\!] \wedge (\sigma',\sigma'') \in f\}
\end{aligned}$$

Substituting `true` in for b give the following equation (which is only true for commands with the same form as c_b):

$$\begin{aligned}
F'(f) &= \{(\sigma,\sigma) \mid (\sigma, \texttt{false}) \in \mathcal{B}[\![\texttt{true}]\!]\} \\
&\cup \ \{(\sigma,\sigma'') \mid (\sigma, \texttt{true}) \in \mathcal{B}[\![\texttt{true}]\!] \wedge (\sigma,\sigma') \in \mathcal{C}[\![c]\!] \wedge (\sigma',\sigma'') \in f\}
\end{aligned}$$

Since $\mathcal{B}[\![\texttt{true}]\!] = \{(\sigma, \texttt{true})\}$, the pair $(\sigma, \texttt{false})$ will never be a subset of $\mathcal{B}[\![\texttt{true}]\!]$ and the set preceding the union operator will be the empty set. Likewise, $(\sigma, \texttt{true})$ is always a subset of $\{(\sigma, \texttt{true})\}$, so we can refine our definition of F' as follows:

$$F'(f) \quad = \quad \{(\sigma,\sigma'') \mid (\sigma,\sigma') \in \mathcal{C}[\![c]\!] \wedge (\sigma',\sigma'') \in f\}$$

Now we'll show that f_n is undefined on all stores for all $n \in \mathbb{N}$ using induction (and the higher order function F'):

$$\begin{aligned} P(n) &\triangleq f_n = \emptyset \\ P(0): &\quad f_0 = F^0(\emptyset) = \emptyset \checkmark \\ P(n+1): &\quad f_{n+1} = F(f_n) \\ &= \{(\sigma, \sigma'') \mid (\sigma, \sigma') \in \mathcal{C}[\![c]\!] \land (\sigma', \sigma'') \in f_n\} \end{aligned}$$

Using our inductive hypothesis, we can conclude that $f_n = \emptyset$ which in turn means that the second rule in the set comprehension (e.g. $(\sigma', \sigma'') \in f_n$) is unsatisfiable. So the right hand side of the equation is just the empty set, as desired:

$$f_{n+1} = \emptyset \checkmark$$

Therefore, $P(n)$ hold for all $n \in \mathbb{N}$ and the fixed point of F' is:

$$\text{fix}(F') = \bigcup_{i \geq 0} F'^i(\emptyset) = \emptyset \cup \emptyset \cup ... \cup \emptyset = \emptyset$$

In short, c_b is guaranteed not to terminate, just like c_a.

(c) **Denotation of** (x := x):

$$\begin{aligned} \mathcal{C}[\![x := x]\!] &= \{(\sigma, \sigma[x \mapsto n]) \mid (\sigma, n) \in \mathcal{A}[\![x]\!]\} \\ \mathcal{A}[\![x]\!] &= \{(\sigma, \sigma)\} \\ \mathcal{C}[\![x := x]\!] &= \{(\sigma, \sigma[x \mapsto \sigma(x)])\} \\ &= \{(\sigma, \sigma)\} \end{aligned}$$

Denotation of ($\texttt{if}(x = x+1)$ $\texttt{then}$ $x := 0$):

$$\mathcal{C}[\![\texttt{if}(x = x+1) \texttt{ then } x := 0]\!] = \{(\sigma, \sigma) \mid (\sigma, \texttt{false}) \in \mathcal{B}[\![x = x+1]\!]\} \cup \{(\sigma, \sigma') \mid (\sigma, \texttt{true}) \in \mathcal{B}$$
$$[\![x = x+1]\!] \land (\sigma, \sigma') \in \mathcal{C}[\![x := 0]\!]\}$$

$$\begin{aligned} \mathcal{A}[\![x]\!] &= \{(\sigma, \sigma)\} \\ \mathcal{C}[\![x := x]\!] &= \{(\sigma, \sigma[x \mapsto \sigma(x)])\} \\ &= \{(\sigma, \sigma)\} \end{aligned}$$

The two are the same.

3. Our loop invariant is: $r = n - qm \land q \geq 0$.

4. (a)

   (b)

5. (a) Yes, consider the following derivation tree (as a side note, we'll refer to $(b \Rightarrow P) \land (\neg b \Rightarrow Q)$ as A:

$$\text{C}\;\dfrac{\vDash \text{A} \Rightarrow \text{A} \qquad \text{W}\;\dfrac{\text{C}\;\dfrac{\vDash (b \Rightarrow P) \land (\neg b \Rightarrow Q) \land b \Rightarrow P \qquad \vdash \{\text{P}\}\ c\ \{\text{A}\} \qquad \vDash A \Rightarrow A}{\vdash \{(b \Rightarrow P) \land (\neg b \Rightarrow Q) \land b\}\ c\ \{A\}}}{\vdash \{\text{A}\}\ \texttt{while b do c}\ \{\text{A} \land \neg b\}} \qquad \text{A} \land \neg b \Rightarrow \text{Q}}{\vdash \{\text{A}\}\ \texttt{while b do c}\ \{\text{Q}\}}$$

*"C" is an abbreviation for "Consequence", and "W" for "While"

Note that we've made a few assumptions here, namely that we have some derivation for {P} c {A} as well as A $\Rightarrow$ A, A $\Rightarrow$ Q, and $(b \Rightarrow P) \land (\neg b \Rightarrow Q) \land b \Rightarrow P$. The last three assumptions are tautologies, however–anything implies itself, and we can combine $(b \Rightarrow P) \land (\neg b \Rightarrow Q)$ with information about the truth value of b to conclude either P (if b is true) or Q (if b is false). Consequently, the entire tree is a proof that

$$\dfrac{\vdash \{P\}\ c\ \{(b \Rightarrow P) \land (\neg b \Rightarrow Q)\}}{\vdash \{(b \Rightarrow P) \land (\neg b \Rightarrow Q)\}\ \texttt{while b do c}\ \{Q\}}$$

which is the definition of our alternate `while` rule.

(b) No, this rule reduces the completeness of Hoare logic. Consider the following code:

```
sum:= 0
i:= 0
while i < n do{
  i:= i + 1
  sum:= sum + i
}
```

It computes $\Sigma_{j=1}^{n} j$, and it would be nice to be able to prove that property using Hoare Logic. Put simply, you can't prove that property using the new rule (primarily because you can only show that the sum is $\Sigma_{j=1}^{i} j$ and that $i <= n + 1$, but not that $i <= n$), but we'll delve into a slightly longer analysis to show where the bug occurs. Here's part of the proof we'd write using the original inference rules (along with a few labels for lengthier subsections of the tree:

<u>LT:</u>

$$\text{Assign}\;\dfrac{}{\vdash \{\texttt{sum} = \Sigma_{j=1}^{i} j \land i \leq n^{i} < n\}\texttt{i:= i + 1}\{\texttt{sum} = \Sigma_{j=1}^{i-1} j \land i - 1 \leq n \land i - 1 < n\}}$$

<u>RT:</u>

$$\text{Assign}\;\dfrac{}{\vdash \{\texttt{sum} = \Sigma_{j=1}^{i-1} j \land i - 1 \leq n \land i - 1 < n\}\texttt{sum:= sum + i}\{\texttt{sum} - i = \Sigma_{j=1}^{i-1} \land i - 1 \leq n \land i - 1 < n\}}$$

<u>A:</u> $\{\texttt{sum} = \Sigma_{j=1}^{i} j \land i \leq n \land i < n\}$

<u>B':</u> $\{\texttt{sum} - i = \Sigma_{j=1}^{i-1} \land i - 1 \leq n \land i - 1 < n\}$

<u>B:</u> $\{\texttt{sum} = \Sigma_{j=1}^{i} j \land i \leq n\}$

Note that you can use arithmetic properties to conclude that B' $\Rightarrow$ B

$$\text{While}\ \dfrac{\text{Consequence}\ \dfrac{\text{anything implies itself}\ \dfrac{}{\text{A} \Rightarrow \text{A}} \qquad \text{Seq}\ \dfrac{\text{LT} \qquad \text{RT}}{\{\text{A}\}\ \texttt{i:=i+1; sum:= sum + i}\ \{\text{B'}\}} \qquad \text{arithmetic}\ \dfrac{}{\text{B'} \Rightarrow \text{B}}}{\{\text{A}\}\ \texttt{i:=i+1; sum:= sum + i}\ \{\text{B}\}}}{\{\texttt{sum} - \Sigma_{j=1}^{i} j \wedge i \leq n\}\ \texttt{while i < n do (i:= i + 1; sum:= sum + i)}\{\text{B} \wedge i \geq n\}}$$

Notice how important it is for $j < n$ to be on the left-side of `i = i+1; sum := sum + i` above the while inference. Using that information we can conclude that `sum` $= \Sigma_{j=1}^{i} j \wedge i \leq n$ after executing the body of the loop (otherwise it might be possible for $i = n + 1$, which would only show that the final value of `sum` to be either $\Sigma_{j=0}^{i} j \wedge (i = n \vee i = n + 1))$. We need to have $\{P \wedge b\}$ at the top-left of our while rule for the language to be as complete as possible.