1 5F-1 Bookkeeping

- **0 pts** Correct

Peer Review ID: 72928381 — enter this when you fill out your peer evaluation via gradescope

Exercise 5F-2. VCGen Do-While As I really never learned how to think about a do-while loop, I'll convert directly from "do c while b" to "c; while b do c".

So do_{inv} c while b becomes:
c; $while_{inv}$ b do c
Let's drop this directly into a VC and follow a similar method as we used for the "vanilla" while rule:

VC(c; $while_{inv}$ b do c,B)

To read this we might say in english: The post condition B holds given the two commands given.
We can drop the while VCGen from the slides in for $while_{inv}$ b do c:
VC(c; INV $\land$($\forall$ x$_1$...x_2. INV $\rightarrow (b \rightarrow (VC(c, INV) \land \neg b \rightarrow B)))$

We have a nested VC, so we can simplify this by splitting on the "and" ($\land$) dividing our first and second commands:

$$VC(c; INV) \land (\forall x_1 ... x_2 . INV \rightarrow (b \rightarrow (VC(c, INV) \land \neg b \rightarrow B))$$

We realized that do-while is just c; while in disguise, so we made like Scooby-Doo and unmasked it!

Exercise 5F-3. VCGen Mistakes I choose Targaryen and Stark because these rules both fail to modify the post condition.

Exercise 5F-3: Targaryen

1. Targaryen

2. A = true
   We don't really care about A.

3. B = (x = 42)
   Some value that should occur when we exit the loop.

4. $\sigma(x) = 0$
   IMP style initialization.

5. $\sigma'(x) = 42$
   The expected final value.

6. c = while $x < 1$ do x := 42
   Some command to intentionally make this fail.

7. $$\frac{\langle x < 1, \sigma \rangle \Downarrow true \quad \langle x := 42, \sigma \rangle \Downarrow \sigma' \quad \langle while \;\; x < 1 \;\; do \;\; x := 42, \sigma' \rangle \Downarrow \sigma''}{\langle while \;\; x < 1 \;\; do \;\; x := 42, \sigma \rangle \Downarrow \sigma''}$$

2 5F-2 VCGen Do-While

- 0 pts Correct

Peer Review ID: 72928381 — enter this when you fill out your peer evaluation via gradescope

Exercise 5F-2. VCGen Do-While As I really never learned how to think about a do-while loop, I'll convert directly from "do c while b" to "c; while b do c".

So do_{inv} c while b becomes:
c; $while_{inv}$ b do c
Let's drop this directly into a VC and follow a similar method as we used for the "vanilla" while rule:

VC(c; $while_{inv}$ b do c,B)

To read this we might say in english: The post condition B holds given the two commands given.
We can drop the while VCGen from the slides in for $while_{inv}$ b do c:
VC(c; INV $\wedge$($\forall$ x$_1$...x_2. INV $\rightarrow (b \rightarrow (VC(c, INV) \wedge \neg b \rightarrow B)))$

We have a nested VC, so we can simplify this by splitting on the "and" ($\wedge$) dividing our first and second commands:

$$VC(c; INV) \wedge (\forall x_1 ... x_2 . INV \rightarrow (b \rightarrow (VC(c, INV) \wedge \neg b \rightarrow B))$$

We realized that do-while is just c; while in disguise, so we made like Scooby-Doo and unmasked it!

Exercise 5F-3. VCGen Mistakes I choose Targaryen and Stark because these rules both fail to modify the post condition.

Exercise 5F-3: Targaryen

1. Targaryen

2. A = true
   We don't really care about A.

3. B = (x = 42)
   Some value that should occur when we exit the loop.

4. $\sigma(x) = 0$
   IMP style initialization.

5. $\sigma'(x) = 42$
   The expected final value.

6. c = while $x < 1$ do x := 42
   Some command to intentionally make this fail.

7. $$\frac{\langle x < 1, \sigma \rangle \Downarrow true \quad \langle x := 42, \sigma \rangle \Downarrow \sigma' \quad \langle while \;\; x < 1 \;\; do \;\; x := 42, \sigma' \rangle \Downarrow \sigma''}{\langle while \;\; x < 1 \;\; do \;\; x := 42, \sigma \rangle \Downarrow \sigma''}$$

If the guard is false do nothing:

$$\frac{\langle x < 1, \sigma \rangle \Downarrow \mathit{false}}{\langle \mathit{while}\ \ x < 1\ \ \mathit{do}\ \ x := 42, \sigma \rangle \Downarrow \sigma}$$

σ'' is our σ'.

8. A is just true, so $\sigma \models A$

9. $\sigma' \models B$ Check this by noticing that the loop does one iteration and ends on x := 42. Therefore $\sigma'(x) = 42$

10. $\vdash$ {A} c {B} is not possible because the rule is assuming pre and post conditions are the same (X = X). We can just check A = B and find true = 42. This is clearly not correct. The rule only works in the case the loop body is never encountered (if b is false coming in).

The rule should consider a correct post condition, and should have the loop guard considered coming in.

Exercise 5F-3: Stark Stark, as usual, is missing some key information. A correct rule probably would have seen the Red Wedding coming. Let's see where the strategy could be improved:

1. Stark

2. A = true
We don't really care about A, it will simply show the modification made in the loop isn't correctly tracked by the post condition.

3. B = (x = 42)
Some value that should occur when we exit the loop.

4. $\sigma(x) = 0$
IMP style initialization.

5. $\sigma'(x) = 42$
The expected final value.

6. c = while $x < 1$ do x := 42
Some command to intentionally make this fail.

7. $$\frac{\langle x < 1, \sigma \rangle \Downarrow \mathit{true} \quad \langle x := 42, \sigma \rangle \Downarrow \sigma' \quad \langle \mathit{while}\ \ x < 1\ \ \mathit{do}\ \ x := 42, \sigma' \rangle \Downarrow \sigma''}{\langle \mathit{while}\ \ x < 1\ \ \mathit{do}\ \ x := 42, \sigma \rangle \Downarrow \sigma''}$$

If the guard is false do nothing:

$$\frac{\langle x < 1, \sigma \rangle \Downarrow \mathit{false}}{\langle \mathit{while}\ \ x < 1\ \ \mathit{do}\ \ x := 42, \sigma \rangle \Downarrow \sigma}$$

note that I add an extra step here to be explicit, but σ'' is our σ'.

8. A is just true, so $\sigma \models A$

9. $\sigma' \models B$ Check this by noticing that the loop does one iteration and ends on x := 42. Therefore $\sigma'(x) = 42$

10. $\vdash$ {A} c {B} is not possible because the rule is assuming pre and post conditions are the same (X = X). We can just check A = B and find true = 42. This is clearly not correct. The rule only works in the case the loop body is never encountered (if b is false coming in).

Lannister seems reasonable but maybe a little needlessly complex.

3 5F-3 VCGen Mistakes

- 0 pts Correct

Peer Review ID: 72928381 — enter this when you fill out your peer evaluation via gradescope