Exercise 5F-2. VCGen Do-While [8 points]. Choose exactly *one* of the two options below. (If you are not certain, pick the first. The answers end up being equivalent, but the first may be easier to grasp for some students and the second easier to grasp for others.)

- Give the (backward) verification condition formula for the command $\textsf{do}_{Inv}\ c\ \textsf{while}\ b$ with respect to a post-condition P. The invariant Inv is true before each evaluation of the predicate b. Your answer may not be defined in terms of VC(while...).
- Give the (backward) verification condition formula for the command $\textsf{do}_{Inv1,Inv2}\ c\ \textsf{while}\ b$ with respect to a post-condition P. The invariant $Inv1$ is true before c is first executed. The invariant $Inv2$ is true before each evaluation of the loop predicate b. Your answer may not be defined in terms of VC(while...).

Solution: We proceed with the first option, and first decompose a do while, into a series of commands:

$$\text{VC}(\textsf{do}_{Inv}\ c\ \textsf{while}\ b, P) = VC(c; \textsf{while}\ _{Inv}\ b\ \textsf{do}\ c, P)$$

Then we apply the VCGen rule for a series of commands:

$$= (c, \text{VC}(\textsf{while}\ _{Inv}\ b\ \textsf{do}\ c, P))$$

Then we apply the VCGen for while

$$= \text{VC}(c, Inv\ \wedge (\forall x_1 \ldots x_n . Inv \implies (e \implies \text{VC}(c, Inv) \wedge \neg e \implies P)))$$

Exercise 5F-3. VCGen Mistakes [20 points]. Consider the following three alternate while Hoare rules (named lannister, stark, and targaryen):

$$\frac{\vdash \{X\}\ c\ \{b \implies X\ \wedge\ \neg b \implies Y\}}{\vdash \{b \implies X\ \wedge\ \neg b \implies Y\}\ \textsf{while}\ b\ \textsf{do}\ c\ \{Y\}}\ \textsf{lannister} \qquad \frac{\vdash \{X\ \wedge\ b\}\ c\ \{X\}}{\vdash \{X\}\ \textsf{while}\ b\ \textsf{do}\ c\ \{X\}}\ \textsf{stark}$$

$$\frac{\vdash \{X\}\ c\ \{X\}}{\vdash \{X\}\ \textsf{while}\ b\ \textsf{do}\ c\ \{X\ \wedge\ \neg b\}}\ \textsf{targaryen}$$

All three rules are sound but incomplete. Choose **two** incomplete rules. For each chosen rule provide the following:

1. the name of the rule and
2. A and
3. B and
4. σ and
5. σ' and
6. c such that
7. $\langle c, \sigma \rangle \Downarrow \sigma'$ and
8. $\sigma \models A$ and
9. $\sigma' \models B$ but
10. it is not possible to prove $\vdash \{A\}\ c\ \{B\}$.

Flavor text: Incompleteness in an axiomatic semantics or type system is typically not as dire as unsoundness. An incomplete system cannot prove all possible properties or handle all possible programs. Many research results that claim to work for the C language, for example, are actually incomplete because they do not address `setjmp/longjmp` or bitfields. (Many of them are also unsound because they do not correctly model unsafe casts, pointer arithmetic, or integer overflow.)

Solution: ‘

1. targaryen
2. $A = (x == 1)||(x == 2)$
3. $B = (x == 1)$
4. $\sigma = (x, 1)$
5. $\sigma' = (x, 1)$
6. $c = x = 1;$ while $x < 1$ do $x = 2$
7. $\langle c, \sigma \rangle \Downarrow \sigma'$ and
8. $\sigma \models A$ and
9. $\sigma' \models B$ but
10. it is not possible to prove $\vdash \{A\}\ c\ \{B\}$.

and

1. stark
2. $A = (x == 0\ ||\ x == 2)$
3. $B = (x == 0\ ||\ x == 2) \wedge (x \geq 1)$
4. $\sigma = (x, 0)$
5. $\sigma' = (x, 2)$
6. $c = x = 0;$ while $x < 1$ do $x = 2$
7. $\langle c, \sigma \rangle \Downarrow \sigma'$ and
8. $\sigma \models A$ and
9. $\sigma' \models B$ but
10. it is not possible to prove $\vdash \{A\}\ c\ \{B\}$.

Submission. Turn in the formal component of the assignment as a single PDF document via the `gradescope` website. Your name and Michigan email address must appear on the first page of your PDF submission but may not appear anywhere else.