Problem 1. Consider the following IMP program.

```
a := 0;
b := 0;
c := 0;
while true {
  assert a >= 0;
  a := a + b;
  b := b + c;
  c := c + 1;
}
```

(a) Manually convert this program into a transition system, where each step of the transition system corresponds to one iteration of the body of the loop. Use some "poetic license" in your conversion, so that the resulting transition system is as simple as possible. (Do not model the first three lines of the program as transitions, but rather as part of your definition of the initial states. Also, ignore the `assert` for now.)

Be sure to clearly define the three components of your transition system: state space, initial states, and step relation.

(b) Convert the assertion into a property P of states in your transition system. Explain briefly and informally why P is an invariant of your transition system.

(c) Explain why a direct proof that P is an invariant by induction on executions fails. (Don't just say "it fails because it's not inductive". Either walk through the start of a proof and point to exactly where you get stuck because the induction hypothesis is not strong enough, or give a concrete counterexample to induction (CTI) that shows P is not inductive and explain why your CTI is, in fact, a CTI.)

(d) Prove P is an invariant indirectly: find a property I that implies P and prove I is an invariant directly by induction on executions. Be sure to clearly state your definition of I. Try to make your I as simple as possible.

(e) Does your invariant I capture exactly the set of reachable states in your transition system?

- If so, prove it by proving that for every state satisfying I, you can construct an execution that reaches it.
- If not, give an example of a state that satisfies I but is not reachable. Prove that your example state is not reachable. (How do we show a state is not reachable...?)

In your answer, be sure to clearly state which of the two possibilities above you are proving. Since there are many invariants I that are inductive and imply P, there are many different answers to this question. That's ok! Just answer the question for the I you picked in the previous part.

Problem 2. This problem is about treating the small-step operational semantics of a language as a transition system.

(a) The small-step operational semantics for IMP are themselves already a giant transition system whose state space consists of all possible pairs of heaps and statements and whose transitions are just steps in the small-step operational semantics.

Define a set of initial states that describes the states at the beginning of an execution of the program from Problem 1. Try to make your definition of the initial states as simple as possible. (Think carefully about what constraints, if any, you need to place on the heap.)

(b) A state in this transition system is a pair of a heap and a (possibly complex) statement. Describe the set of *statements* that can appear in reachable states, ignoring the heap. No need to prove that your answer is correct just yet. (Hint: There are more than 5 and fewer than 15. It's ok not to list them out explicitly, but to describe them succinctly.)

(c) Are there finitely many *heaps* that can appear in reachable states? Give a one-sentence proof of your yes/no answer.

(d) Using the "official" version of the semantics for IMP (i.e., the one with `failed`, which is reproduced at the end of this document), we can encode the English statement

> The assertion in the program from Problem 1 is never violated.

by saying that the statement `failed` does not appear in any reachable state. State this formally using metavariables and symbols. Call this property P.

(e) Explain informally why P is an invariant of your transition system.

(f) Explain why a direct proof that P is an invariant by induction on executions fails. (Don't just say "it fails because it's not inductive". Either walk through the start of a proof and point to exactly where you get stuck because the induction hypothesis is not strong enough, or, give a concrete CTI and explain why your CTI is, in fact, a CTI.)

(g) Prove P is an invariant indirectly: find a property I that implies P and prove I is an invariant directly by induction on executions. Be sure to clearly state your definition of I.

(h) Explain why your I in this problem is more complicated than the I from Problem 1. (Hint: What is the "granularity" of a step in the transition system from Problem 1? What about in the transition system in this problem? What is the state space in Problem 1? What about in this problem?)

(i) Does your invariant I capture exactly the set of *statements* that can appear in reachable states, as you enumerated in part (b)?

- If so, prove it by proving that for every statement that can appear in a state satisfying I, you can construct an execution that reaches a state containing that statement.
- If not, give an example of a statement that can appear in a state satisfying I but not in any reachable state. Prove that your example statement cannot appear in any reachable state. (How do we show something about all reachable states...?)

In your answer, be sure to clearly state which of the two possibilities above you are proving. Since there are many invariants I that are inductive and imply P, there are many different answers to this question. That's ok! Just answer the question for the I you picked in part (g).

Problem 3. Consider the transition system from lecture that we produced by manually translating our favorite program into a transition system. We reproduce the definition here. The transition system is parameterized by an integer n.

$$\begin{aligned} S &= \{(x, y) \mid x, y \in \mathbb{Z}\} \\ S_0 &= \{(0, n)\} \\ \to &= \left\{ \big((x, y), (x+1, y-1)\big) \,\middle|\, y > 0 \right\} \end{aligned}$$

(a) Which states cannot step? Give your answer as a definition of a set of states, and a one-sentence proof that a state cannot step if and only if it is in your set.

$$S = \{(x, y) \mid x, y \in \mathbb{Z} \wedge y \leqslant 0\}$$

States in this set cannot step, because according to our definition of step, a state can step if and only if $y > 0$. However, in this set of states, the y values are less or equal to 0 thus cannot step.

(b) Define a set of "final" states, which are those we intuitively expect execution to end in. There is more than one reasonable answer here, so pick whatever seems most intuitive to you, subject to your ability to prove the next part below.

$$S = \{(x, 0) \mid x \in \mathbb{Z} \wedge x \geqslant 0\}$$

(c) Prove that for all $n \in \mathbb{Z}$, if $n \geqslant 0$ then there exists a final state s such that $(0, n) \to^* s$. (Hint: There is more than one thing you could try to prove by induction, but really only one choice of *what* to induct on.)

Lemma 1: for all $x, y \in \mathbb{N}$, $(x, y) \to^* s$ for some arbitary but fixed final state s

Proof. By induction on y.

- $y = 0$. In this case, by the reflexivity of step and $\to^*$, $(x, 0) \to^* (x, 0)$.
- $y = y' + 1$. According to the definition of step, $(x, y' + 1) \to (x + 1, y')$. Since $x \in \mathbb{N}$ and $y' \in \mathbb{N}$, $x + 1 \in \mathbb{N}$, and thus according to the induction hypothesis, $(x + 1, y') \to^* s$ for some arbitary but fixed final state s. According to the definition of $\to^*$,

$$\frac{(x, y' + 1) \to (x + 1, y') \qquad (x + 1, y') \to^* s}{(x, y' + 1) \to^* s}$$

for some arbitary but fixed final state s.

□

Theorem: for all $n \in \mathbb{Z}$, if $n \geqslant 0$ then there exists a final state s such that $(0, n) \to^* s$

Proof. Since $0 \in \mathbb{N}$, and $n \in \mathbb{Z} \wedge n \geqslant 0 \leftrightarrow n \in \mathbb{N}$, according to *Lemma 1*, $(0, n) \to^* s$ for some fixed final state s. □

(d) Explain how your proof from the previous part, together with the fact that the transition system is deterministic, also shows that there are no infinite executions in this transition system.

According to the proof above, for all initial state $s \in \mathbb{S}_0$, there exists a final state s' such that $s \to^* s'$. Since this transition system is deterministic, there exists exactly one such final state s' that $s \to^* s'$. So for all $s \in \mathbb{S}_0$, the execution starts at s terminates at a unique final state s'.

(e) Construct an infinite execution of your transition system from Problem 1.

The transition system is

$$\begin{aligned} S &= \{(a, b, c) \mid x, y \in \mathbb{Z}\} \\ S_0 &= \{(0, 0, 0)\} \\ \to &= \left\{\big((a, b, c), (a + b, b + c, c + 1)\big) \;\middle|\; a \geqslant 0\right\} \end{aligned}$$

Frist, prove that the transition system does have infinite execution by showing that $a \geqslant 0$ holds for all reachable states. We choose a strenghened invariant: $a, b, c \in \mathbb{Z} \wedge a, b, c \geqslant 0$.

Proof. By induction on the states.

i. $s = (0, 0, 0)$. The initial state satisfies the condition that $a, b, c \in \mathbb{Z} \wedge a, b, c \geqslant 0$.

ii. $(a, b, c) \to (a' = a + b, b' = b + c, c' = c + 1)$. The induction hypothesis tells us that $a, b, c \in \mathbb{Z} \wedge a, b, c \geqslant 0$. Therefore, $a' \geqslant a + b \wedge b' \geqslant b + c \wedge c' > c$. Therefore, $a', b', c' \in \mathbb{Z} \wedge a' \geqslant 0 \wedge b' \geqslant 0 \wedge c' \geqslant 0$.

□

The infinite execution is:

$$(0, 0, 0) \to (0, 0, 1) \to (0, 1, 2) \to (1, 3, 3) \to (4, 6, 4) \to (10, 10, 5) \cdots$$

Problem 4. Consider the "official" version of Imp, with `failed` (full semantics on next page). In Lecture 6, we stated the following progress lemma:

> If $\vdash_\Sigma s$ ok and $\vdash_\Sigma H$ ok, then $s =$ `skip` or $s =$ `failed` or there exists H', s' such that $H, s \to H', s'$

(a) Enumerate all the things you *could* try to induct on. For each one, say whether you think the proof would work if you inducted on that thing. (As usual, it's ok to be wrong.)

(b) Prove the progress lemma by induction on a thing of your choice. As we mentioned in class, you will need at least one lemma. Try to make your lemma(s) as simple as possible. You do not need to prove any additional lemmas other than progress, just state what you're using clearly (and convince yourself it's true!).

IMP

$$
\begin{array}{lcl}
e & ::= & n \mid e + e \mid e - e \mid b \mid e < e \mid e = e \mid x \\
v & ::= & n \mid b \\
s & ::= & \texttt{skip} \mid \texttt{assert}\ e \mid \texttt{failed} \mid x := e \mid s; s \mid \texttt{while}\ e\ s \\
\tau & ::= & \text{int} \mid \text{bool}
\end{array}
\qquad
\begin{array}{lcl}
n & \in & \mathbb{Z} \\
b & \in & \mathbb{B} \\
x & \in & \text{Var}\ (= \text{String}) \\
H & \in & \text{Var} \rightharpoonup \text{Value} \\
\Sigma & \in & \text{Var} \rightharpoonup \text{Type}
\end{array}
$$

$\boxed{H, e \Downarrow v}$

$$\frac{}{H, n \Downarrow n} \qquad \frac{}{H, b \Downarrow b} \qquad \frac{x \in \text{dom}\ H \qquad H(x) = v}{H, x \Downarrow v}$$

$$\frac{H, e_1 \Downarrow n_1 \qquad H, e_2 \Downarrow n_2}{H, e_1 + e_2 \Downarrow n_1 + n_2} \qquad \frac{H, e_1 \Downarrow n_1 \qquad H, e_2 \Downarrow n_2}{H, e_1 - e_2 \Downarrow n_1 - n_2}$$

$$\frac{H, e_1 \Downarrow n_1 \qquad H, e_2 \Downarrow n_2}{H, e_1 < e_2 \Downarrow n_1 < n_2} \qquad \frac{H, e_1 \Downarrow v_1 \qquad H, e_2 \Downarrow v_2}{H, e_1 = e_2 \Downarrow n_1 = n_2}$$

$\boxed{H, s \to H, s}$

$$\frac{H, e \Downarrow v}{H, x := e \to H[x \mapsto v], \texttt{skip}} \qquad \frac{H, e \Downarrow \top}{H, \texttt{assert}\ e \to H, \texttt{skip}} \qquad \frac{H, e \Downarrow \bot}{H, \texttt{assert}\ e \to H, \texttt{failed}}$$

$$\frac{H, s_1 \to H', s_1'}{H, s_1; s_2 \to H', s_1'; s_2} \qquad \frac{}{H, \texttt{skip}; s \to H, s} \qquad \frac{}{H, \texttt{failed}; s \to H, \texttt{failed}}$$

$$\frac{H, e \Downarrow \top}{H, \texttt{while}\ e\ s \to H, s; \texttt{while}\ e\ s} \qquad \frac{H, e \Downarrow \bot}{H, \texttt{while}\ e\ s \to H, \texttt{skip}}$$

$\boxed{\vdash_\Sigma e : \tau}$

$$\frac{}{\vdash_\Sigma n : \text{int}} \qquad \frac{}{\vdash_\Sigma b : \text{bool}} \qquad \frac{x \in \text{dom}\ \Sigma \qquad \Sigma(x) = \tau}{\vdash_\Sigma x : \tau}$$

$$\frac{\vdash_\Sigma e_1 : \text{int} \qquad \vdash_\Sigma e_2 : \text{int}}{\vdash_\Sigma e_1 + e_2 : \text{int}} \qquad \frac{\vdash_\Sigma e_1 : \text{int} \qquad \vdash_\Sigma e_2 : \text{int}}{\vdash_\Sigma e_1 - e_2 : \text{int}}$$

$$\frac{\vdash_\Sigma e_1 : \text{int} \qquad \vdash_\Sigma e_2 : \text{int}}{\vdash_\Sigma e_1 < e_2 : \text{bool}} \qquad \frac{\vdash_\Sigma e_1 : \tau \qquad \vdash_\Sigma e_2 : \tau}{\vdash_\Sigma e_1 = e_2 : \text{bool}}$$

$\boxed{\vdash_\Sigma s\ \text{ok}}$

$$\frac{}{\vdash_\Sigma \texttt{skip}\ \text{ok}} \qquad \frac{x \in \text{dom}\ \Sigma \qquad \vdash_\Sigma e : \Sigma(x)}{\vdash_\Sigma x := e\ \text{ok}} \qquad \frac{\vdash_\Sigma e : \text{bool}}{\vdash_\Sigma \texttt{assert}\ e\ \text{ok}}$$

$$\frac{\vdash_\Sigma s_1\ \text{ok} \qquad \vdash_\Sigma s_2\ \text{ok}}{\vdash_\Sigma s_1; s_2\ \text{ok}} \qquad \frac{\vdash_\Sigma e : \text{bool} \qquad \vdash_\Sigma s\ \text{ok}}{\vdash_\Sigma \texttt{while}\ e\ s\ \text{ok}}$$

$\boxed{\vdash_\Sigma H\ \text{ok}}$

$$\frac{\forall x \in \text{dom}\ \Sigma,\ x \in \text{dom}\ H\ \wedge \vdash_\Sigma H(x) : \Sigma(x)}{\vdash_\Sigma H\ \text{ok}}$$