Solutions 3

Exercise 1.

Axiom:	$\forall x.\, E(x,x)$
NNF:	$\forall x.\, E(x,x)$
Skolemization:	$\forall x.\, E(x,x)$
PNF:	$E(x,x)$

Axiom:	$\forall x,y,z.\, (P(x,y) \land P(y,z)) \to P(x,z)$
NNF:	$\forall x,y,z.\, \neg P(x,y) \lor \neg P(y,z) \lor P(x,z)$
Skolemization:	$\forall x,y,z.\, \neg P(x,y) \lor \neg P(y,z) \lor P(x,z)$
PNF:	$\neg P(x,y) \lor \neg P(y,z) \lor P(x,z)$

Axiom:	$\forall x,y.\, P(x,y) \to \exists z.\, P(x,z) \land f(z,y)$
NNF:	$\forall x,y.\, \neg P(x,y) \lor \exists z.\, P(x,z) \land f(z,y)$
Skolemization:	$\forall x,y.\, \neg P(x,y) \lor P(x, f_z(x,y)) \land f(f_z(x,y),y)$
PNF:	$\neg P(x,y) \lor P(x, f_z(x,y)) \land f(f_z(x,y),y)$

Axiom:	$\forall x,y.\, E(x,y) \leftrightarrow \neg(P(x,y) \lor P(y,z))$
NNF:	$\forall x,y.\, (\neg E(x,y) \lor \neg P(x,y) \lor \neg P(y,z)) \land (P(x,y) \lor P(y,z) \lor E(x,y))$
Skolemization:	$\forall x,y.\, (\neg E(x,y) \lor \neg P(x,y) \lor \neg P(y,z)) \land (P(x,y) \lor P(y,z) \lor E(x,y))$
PNF:	$(\neg E(x,y) \lor \neg P(x,y) \lor \neg P(y,z)) \land (P(x,y) \lor P(y,z) \lor E(x,y))$

Axiom:	$\forall x,y,z.\, (E(x,y) \land E(y,z)) \to E(x,z)$
NNF:	$\forall x,y,z.\, \neg E(x,y) \lor \neg E(y,z) \lor E(x,z)$
Skolemization:	$\forall x,y,z.\, \neg E(x,y) \lor \neg E(y,z) \lor E(x,z)$
PNF:	$\neg E(x,y) \lor \neg E(y,z) \lor E(x,z)$

Axiom:	$\forall x,y,z.\, (E(x,y) \land P(x,z)) \to P(y,z)$
NNF:	$\forall x,y,z.\, \neg E(x,y) \lor \neg P(x,z) \lor P(y,z)$
Skolemization:	$\forall x,y,z.\, \neg E(x,y) \lor \neg P(x,z) \lor P(y,z)$
PNF:	$\neg E(x,y) \lor \neg P(x,z) \lor P(y,z)$

Axiom:	$\forall x, y, z.\, (E(x, y) \wedge P(z, x)) \rightarrow P(z, y)$
NNF:	$\forall x, y, z.\, \neg E(x, y) \vee \neg P(z, x) \vee P(z, y)$
Skolemization:	$\forall x, y, z.\, \neg E(x, y) \vee \neg P(z, x) \vee P(z, y)$
PNF:	$\neg E(x, y) \vee \neg P(z, x) \vee P(z, y)$

Herbrand's theorem tells us that we can find a countable dense order. $(\mathbb{Q}, <, =)$ is such an example.

Exercise 2.

1. The most complicated part is finding a procedure to find an formula equisat to $\neg F_1$.

   $$\neg\forall x_1 ... \forall x_n.\ F(x_1, ..., x_n) \equiv \exists x_1 ... \exists x_n.\ \neg F(x_1, ..., x_n)$$

   By applying Skolemization, we can remove the first quantifier and obtain the following equisatisfiable formula:

   $$\exists x_2 ... \exists x_n.\ \neg F(f_{x_1}, ..., x_n)$$

   Where f_{x_1} is a new constant symbol introduced by the skolemization. By repeating the process other $n - 1$ times we can prove by induction that $\neg F$ is equisat to:

   $$\neg F(f_{x_1}, ..., f_{x_n})$$

   which is quantifier free and whose signature contains only constant symbols and predicates.

   The disjunction of $\forall x_1 ... \forall x_n.\ F_1(x_1, ..., x_n)$ and $\forall y_1 ... \forall y_m.\ F_2(y_1, ..., y_m)$ is equisatisfiable to $\forall x_1 ... \forall x_n . \forall y_1 ... \forall y_m.\ F_1(x_1, ..., x_n) \vee F_2(y_1, ..., y_m)$. The same goes for conjunction.

2. First, note that since the theory is composed of one single formula (which has a finite size), one can assume that the set of relations $\mathcal{R}$ and constants $\mathcal{C}$ of the language is finite (i.e. restricted to the ones appearing in the formula at hand). Without loss of generality, suppose the language contains at least one constant symbol (possibly not appearing in the formula).

   Suppose the formula is satisfiable, i.e. it has a model. Then, Herbrand's theorem applies and $(\mathrm{GT}_{\mathcal{L}}, \alpha_h)$ is also a model. However, since all functions have arity 0 (i.e. are constants), we have $\mathrm{GT}^i = \mathrm{GT}^j$ for any $i, j \in \mathbb{N}^*$. This means that

$$\mathrm{GT}_{\mathcal{L}} = \bigcup_{i \geq 0} \mathrm{GT}^i = \mathrm{GT}^1 = \mathcal{C}$$

   Since the set of constants is finite, so is the domain of the ground model. In particular, the formula has a model if and only if it has a model of size $|\mathcal{C}|$.

   Thus, it is sufficient to try exhaustively all possible interpretations of the relation symbols on this finite domain. There are exponentially, but finitely, many different such interpretations.

   To summarize the procedure:

   (a) Generate the finite ground domain of the language;

   (b) For each possible interpretation of the relation symbols, evaluate the formula. The formula is satisfiable if and only if is evaluates to true on at least one interpretation.

   As soon as the formula holds for all assignments in one of the interpretations, the formula is satisfiable; else, if it does in none, it is unsatisfiable.

Exercise 3.

- The first statement is false:

  Let $r_1 = \{(a, a), (b, a)\}$, $r_2 = \{(a, b), (b, b)\}$ and $Q = \{b\}$.

  Then $\mathrm{wp}(r_1, Q) = \emptyset$, $\mathrm{wp}(r_2, Q) = \{a, b\}$ and $\mathrm{wp}(r_1 \cup r_2, Q) = \emptyset$.

  Note that r_1, r_2 and $r_1 \cup r_2$ are all functional relations.

- The second statement is true:

$$\begin{aligned}
\mathrm{wp}(r_1 \cup r_2, Q) &= \{s \mid \forall s'.\ (s, s') \in r_1 \cup r_2 \to s' \in Q\} \\
&= \{s \mid \forall s'.\ (s, s') \in r_1 \vee (s, s') \in r_2 \to s' \in Q\} \\
&= \{s \mid \forall s'.\ (s, s') \in r_1 \to s' \in Q \wedge (s, s') \in r_2 \to s' \in Q\} \\
&= \{s \mid (\forall s'.\ (s, s') \in r_1 \to s' \in Q) \wedge (\forall s'.\ (s, s') \in r_2 \to s' \in Q)\} \\
&= \{s \mid \forall s'.\ (s, s') \in r_1 \to s' \in Q\} \cap \{s \mid \forall s'.\ (s, s') \in r_2 \to s' \in Q\} \\
&= \mathrm{wp}(r1, Q) \cap \mathrm{wp}(r2, Q)
\end{aligned}$$

- The third statement is also true:

$$\begin{aligned}
\mathrm{wp}(r, Q_1 \cap Q_2) &= \{s \mid \forall s'.\ (s, s') \in r \to s' \in Q_1 \cap Q_2\} \\
&= \{s \mid \forall s'.\ (s, s') \in r \to (s' \in Q_1 \wedge s' \in Q_2)\} \\
&= \{s \mid \forall s'.\ ((s, s') \in r \to s' \in Q_1) \wedge ((s, s') \in r \to s' \in Q_2)\} \\
&= \{s \mid (\forall s'.\ (s, s') \in r \to s' \in Q_1) \wedge (\forall s'.\ (s, s') \in r \to s' \in Q_2)\} \\
&= \{s \mid \forall s'.\ (s, s') \in r \to s' \in Q_1\} \wedge \{s \mid \forall s'.\ (s, s') \in r \to s' \in Q_2\} \\
&= \mathrm{wp}(r, Q_1) \cap \mathrm{wp}(r, Q_2)
\end{aligned}$$

- The fourth statement is true only when r is functional:

  Let $r = \{(a, b), (a, c)\}, Q_1 = \{b\}, Q_2 = \{c\}$.

  We have $\mathrm{wp}(r, Q_1) = \mathrm{wp}(r, Q_2) = \{b, c\}$ but $\mathrm{wp}(r, Q_1 \cup Q_2) = \{a, b, c\}$.

  When r is a partial function the weakest precondition becomes the union between the preimage and the set of all the elements that are not in the domain. This is due to the fact that when r is a partial function the image of an element is either a singleton or the empty set. Let s be an arbitrary element, if s is not in the domain then

$$\forall s'.\ (s, s') \in r \to s' \in Q$$

  is true as the condition of the implication is always false. If s is in the domain then the statement reduces to whether its image is in Q, which is equivalent to $\exists s'.\ (s, s') \in r \wedge s' \in Q$. The set of elements verifying this property is the preimage of Q under r also noted $r^{-1}[Q]$. Therefore we have $\mathrm{wp}(r, Q) = r^{-1}[Q] \cup \{s \mid \forall s'.\ (s, s') \notin r\}$

We first show that the preimage of an union is the union of the preimages.

$$\begin{aligned}
r^{-1}[Q_1 \cup Q_2] &= \{s \mid \exists s'.\ (s, s') \in r \wedge s' \in Q_1 \cup Q_2\} \\
&= \{s \mid \exists s'.\ (s, s') \in r \wedge (s' \in Q_1 \vee s' \in Q_2)\} \\
&= \{s \mid \exists s'.\ ((s, s') \in r \wedge s' \in Q_1) \vee ((s, s') \in r \wedge s' \in Q_2)\} \\
&= \{s \mid (\exists s'.\ (s, s') \in r \wedge s' \in Q_1) \vee (\exists s'.\ (s, s') \in r \wedge s' \in Q_2)\} \\
&= \{s \mid \exists s'.\ (s, s') \in r \wedge s' \in Q_1\} \cup \{s \mid \exists s'.\ (s, s') \in r \wedge s' \in Q_2\} \\
&= r^{-1}[Q_1] \cup r^{-1}[Q_2]
\end{aligned}$$

We thus have:

$$\begin{aligned}
\mathrm{wp}(r, Q_1 \cup Q_2) &= r^{-1}[Q_1 \cup Q_2] \cup \{s \mid \forall s'.\ (s, s') \notin r\} \\
&= r^{-1}[Q_1] \cup r^{-1}[Q_2] \cup \{s \mid \forall s'.\ (s, s') \notin r\} \\
&= \mathrm{wp}(r, Q_1) \cup \mathrm{wp}(r, Q_2)
\end{aligned}$$

The last step uses the fact that the set of elements which are not in the domain do not depend on Q_1 or Q_2

Alternative proof when r is functional is as follows. First, observe that, in general

$$wp(r, Q) = \{s \mid r[\{s\}] \subseteq Q\}$$

When r is functional then each $r[\{s\}]$ is either empty set or a singleton set. Next, when C is an empty or singleton sets, then

$$C \subseteq A \cup B \quad \Longleftrightarrow \quad (C \subseteq A \ \vee \ C \subseteq B)$$

Putting all these together, we have that, for functional r,

$$\begin{aligned}
wp(r, Q_1 \cup Q_2) &= \{s \mid r[\{s\}] \subseteq Q_1 \cup Q_2\} \\
&= \{s \mid r[\{s\}] \subseteq Q_1 \vee r[\{s\}] \cup Q_2\} \\
&= \{s \mid r[\{s\}] \subseteq Q_1\} \cup \{s \mid r[\{s\}] \subseteq Q_2\} \\
&= wp(r, Q_1) \cup wp(r, Q_2)
\end{aligned}$$

Exercise 4.

- Using quite mechanically the following:

Command	Formula
f; s	$\exists x_i, y_i.\ R(f)\,[x', y' := x_i, y_i] \wedge R(s)\,[x, y := x_i, y_i]$
if c **then** t **else** e	$(R(c) \wedge R(t)) \vee (\neg R(c) \wedge R(e))$

  We get:

Command	Formula
x = x+y	$F_1 := \quad x' = x + y \wedge y' = y$
y = x−y	$F_2 := \quad x' = x \wedge y' = x - y$
x = x−y	$F_3 := \quad x' = x - y \wedge y' = y$
y = y−x	$F_4 := \quad x' = x \wedge y' = y - x$
if body	$F_{\text{if}} := \quad \exists x_{i1}, y_{i1}, x_{i2}, y_{i2}.$ $F_1\,[x', y' := x_{i1}, y_{i1}] \wedge$ $F_2\,[x, y, x', y' := x_{i1}, y_{i1}, x_{i2}, y_{i2}] \wedge$ $F_3\,[x, y := x_{i2}, y_{i2}]$
else body	$F_{\text{else}} := \quad F_4$
program	$F := \quad ((x > y) \wedge F_{\text{if}}) \vee (\neg(x > y) \wedge F_{\text{else}})$

- We first express the pre- and post-condition:

$$P = x > 0 \wedge y > 0$$
$$Q = 2x + y > 2x' + y' \wedge x' > 0 \wedge y' > 0$$

  We can then express the correctness formula:

$$\forall x, y, x', y'.\ (P \wedge F) \rightarrow Q$$

  One can conclude that this function is correct by realizing that the function amounts to:

  - If $x > y$, swap x and y;
  - Else, decrease y by x.

  and then doing case analysis.

- We note $\oplus$ for wrapping addition (e.g. $(2^{31} - 1) \oplus 1 = -2^{31}$).

  The body of the program is still correct, however the postcondition doesn't hold: consider $x = 1,\ y = 2^{31} - 1$. Then $x' = 1,\ y' = 2^{31} - 2$. The first postcondition is

$$2^{31} \oplus 1 = 2 \oplus (2^{31} - 1) = 2x \oplus y \overset{?}{>} 2x' \oplus y' = 2 \oplus (2^{31} - 2) = 2^{31}$$

However, due to wrapping, $2^{31} \oplus 1 = -2^{31} < 0$.

If you want to test this with stainless, you have to use ––strict–arithmetic=false.