Appendix C

Answers to Selected Exercises

1.1 The following is a possible solution (but not the only one):

$$\begin{gathered}
\forall X(\mathit{natural}(X) \supset \exists Y(\mathit{equal}(s(X), Y))) \\
\neg\exists X\, \mathit{better}(X, \mathit{taking_a_nap}) \\
\forall X(\mathit{integer}(X) \supset \neg \mathit{negative}(X)) \\
\forall X, Y(\mathit{name}(X, Y) \wedge \mathit{innocent}(X) \supset \mathit{changed}(Y)) \\
\forall X(\mathit{area_of_cs}(X) \supset \mathit{important_for}(\mathit{logic}, X)) \\
\forall X(\mathit{renter}(X) \wedge \mathit{in_accident}(X) \supset \mathit{pay_deductible}(X))
\end{gathered}$$

1.2 The following is a possible solution (but not the only one):

$$\begin{gathered}
\mathit{better}(\mathit{bronze_medal}, \mathit{nothing}) \\
\neg\exists X\, \mathit{better}(X, \mathit{gold_medal}) \\
\mathit{better}(\mathit{bronze_medal}, \mathit{gold_medal})
\end{gathered}$$

1.3 Let $\mathrm{MOD}(X)$ denote the set of all models of the formulas X. Then:

$$\begin{array}{rll}
P \models F & \text{iff} & \mathrm{MOD}(P) \subseteq \mathrm{MOD}(F) \\
& \text{iff} & \mathrm{MOD}(P) \cap \mathrm{MOD}(\neg F) = \varnothing \\
& \text{iff} & \mathrm{MOD}(P \cup \{\neg F\}) = \varnothing \\
& \text{iff} & P \cup \{\neg F\} \text{ is unsatisfiable}
\end{array}$$

1.4 Take for instance, $F \supset G \equiv \neg F \vee G$. Let $\Im$ and φ be an arbitrary interpretation and valuation respectively. Then:

$$\begin{array}{rll}
\Im \models_\varphi F \supset G & \text{iff} & \Im \models_\varphi G \text{ whenever } \Im \models_\varphi F \\
& \text{iff} & \Im \models_\varphi G \text{ or } \Im \not\models_\varphi F \\
& \text{iff} & \Im \models_\varphi \neg F \text{ or } \Im \models_\varphi G \\
& \text{iff} & \Im \models_\varphi \neg F \vee G
\end{array}$$

1.6 Let MOD(X) denote the set of all models of the formula X. Then:

$$\begin{array}{lll} F \equiv G & \text{iff} & \mathrm{MOD}(F) = \mathrm{MOD}(G) \\ & \text{iff} & \mathrm{MOD}(F) \subseteq \mathrm{MOD}(G) \text{ and } \mathrm{MOD}(G) \subseteq \mathrm{MOD}(F) \\ & \text{iff} & \{F\} \models G \text{ and } \{G\} \models F \end{array}$$

1.8 Hint: Assume that there is a finite interpretation and establish a contradiction using the semantics of formulas.

1.12 Hints:

$E(\theta\sigma) = (E\theta)\sigma$: by the definition of application it suffices to consider the case when E is a variable.

$(\theta\sigma)\gamma = \theta(\sigma\gamma)$: it suffices to show that the two substitutions give the same result when applied to an arbitrary variable. The fact that $E(\theta\sigma) = (E\theta)\sigma$ can be used to complete the proof.

1.15 Only the last one. (Look for counter-examples of the first two!)

2.1 Definite clauses:

$$\begin{array}{ll} (1) & p(X) \leftarrow q(X). \\ (2) & p(X) \leftarrow q(X, Y), r(X). \\ (3) & r(X) \leftarrow p(X), q(X). \\ (4) & p(X) \leftarrow q(X), r(X). \end{array}$$

2.3 The Herbrand universe:

$$U_P = \{a, b, f(a), f(b), g(a), g(b), f(g(a)), f(g(b)), f(f(a)), f(f(b)), \ldots\}$$

The Herbrand base:

$$B_P = \{q(x, y) \mid x, y \in U_P\} \cup \{p(x) \mid x \in U_P\}$$

2.4 $U_P = \{0, s(0), s(s(0)), \ldots\}$ and $B_P = \{p(x, y, z) \mid x, y, z \in U_P\}$.

2.5 Formulas 2, 3 and 5. Hint: Consider ground instances of the formulas.

2.6 Use the immediate consequence operator:

$$\begin{array}{lll} T_P \uparrow 0 & = & \varnothing \\ T_P \uparrow 1 & = & \{q(a, g(b)), q(b, g(b))\} \\ T_P \uparrow 2 & = & \{p(f(b))\} \cup T_P \uparrow 1 \\ T_P \uparrow 3 & = & T_P \uparrow 2 \end{array}$$

That is, $M_P = T_P \uparrow 3$.

2.7 Use the immediate consequence operator:

$$\begin{array}{lll} T_P \uparrow 0 & = & \varnothing \\ T_P \uparrow 1 & = & \{p(0, 0, 0), p(0, s(0), s(0)), p(0, s(s(0)), s(s(0))), \ldots\} \\ T_P \uparrow 2 & = & \{p(s(0), 0, s(0)), p(s(0), s(0), s(s(0))), \ldots\} \cup T_P \uparrow 1 \\ & \vdots & \\ T_P \uparrow \omega & = & \{p(s^x(0), s^y(0), s^z(0)) \mid x + y = z\} \end{array}$$

3.1 $\{X/a, Y/a\}$, not unifiable, $\{X/f(a), Y/a, Z/a\}$ and the last pair is not unifiable because of occur-check.

3.2 Let σ be a unifier of s and t. By the definition of mgu there is a substitution δ such that $\sigma = \theta\delta$. Now since ω is a renaming it follows also that $\sigma = \theta\omega\omega^{-1}\delta$. Thus, for every unifier σ of s and t there is a substitution $\omega^{-1}\delta$ such that $\sigma = (\theta\omega)(\omega^{-1}\delta)$.

3.4 Assume that σ is a unifier of s and t. Then by definition $\sigma = \theta\omega$ for some substitution ω. Moreover, $\sigma = \theta\theta\omega$ since θ is idempotent. Thus, $\sigma = \theta\sigma$.

Next, assume that $\sigma = \theta\sigma$. Since θ is an mgu it must follow that σ is a unifier.

3.5 $\{X/b\}$ is produced twice and $\{X/a\}$ once.

3.6 For instance, the program and goal:

$$\begin{array}{l} \leftarrow p. \\ p \leftarrow p, q. \end{array}$$

Prolog's computation rule produces an infinite tree whereas a computation rule which always selects the rightmost subgoal yields a finitely failed tree.

3.7 Infinitely many. But there are only two answers, $X = b$ and $X = a$.

4.4 Hint: Each clause of the form:

$$p(t_1, \ldots, t_m) \leftarrow B$$

in P gives rise to a formula of the form:

$$p(X_1, \ldots, X_m) \leftrightarrow \ldots \vee \exists \ldots (X_1 = t_1, \ldots, X_m = t_m, B) \vee \ldots$$

in $comp(P)$. Use truth-preserving rewritings of this formula to obtain the program clause.

4.7 Only P_1 and P_3.

4.8 $comp(P)$ consists of:

$$\begin{array}{l} p(X_1) \leftrightarrow X_1 = a, \neg q(b) \\ q(X_1) \leftrightarrow \Box \end{array}$$

and some equalities including $a = a$ and $b = b$.

4.14 The well-founded model is $\{r, \neg s\}$.

5.1 Without cut there are seven answers. Replacing $true(1)$ by cut eliminates the answers $X = e, Y = c$ and $X = e, Y = d$. Replacing $true(2)$ by cut eliminates in addition $X = b, Y = c$ and $X = b, Y = d$.

5.3 The goal without negation gives the answer $X = a$ while the other goal succeeds without binding X.

5.4 For example:

$$var(X) \leftarrow not(not(X = a)), not(not(X = b)).$$

5.6 For example:

$$between(X, Z, Z) \leftarrow X \leq Z.$$
$$between(X, Y, Z) \leftarrow X < Z, W \ is \ Z - 1, between(X, Y, W).$$

5.7 For instance, since $(n + 1)^2 = n^2 + 2 * n + 1$, $n \geq 0$:

$$sqr(0, 0).$$
$$sqr(s(X), s(Z)) \leftarrow sqr(X, Y), times(s(s(0)), X, W), plus(Y, W, Z).$$

5.8 For instance:

$$gcd(X, 0, X) \leftarrow X > 0.$$
$$gcd(X, Y, Z) \leftarrow Y > 0, W \ is \ X \ mod \ Y, gcd(Y, W, Z).$$

6.2 For instance:

$$grandchild(X, Z) \leftarrow parent(Y, X), parent(Z, Y).$$

$$sister(X, Y) \leftarrow female(X), parent(Z, X), parent(Z, Y), X \neq Y.$$

$$brother(X, Y) \leftarrow male(X), parent(Z, X), parent(Z, Y), X \neq Y.$$

etc.

6.3 HINT: (1) Colours should be assigned to countries. Hence, represent the countries by variables. (2) Describe the map in the goal by saying which countries should be assigned different colours.

6.4 For instance:

$$\begin{array}{l} and(1, 1, 1). \\ and(0, 1, 0). \\ and(1, 0, 0). \\ and(0, 0, 0). \\ \\ inv(1, 0). \\ inv(0, 1). \\ \\ circuit1(X, Y, Z) \leftarrow \\ \qquad and(X, Y, W), inv(W, Z). \\ circuit2(X, Y, Z, V, W) \leftarrow \\ \qquad and(X, Y, A), and(Z, V, B), \\ \qquad and(A, B, C), inv(C, W). \end{array}$$

6.5 For instance:

$$p(X, Y) \leftarrow husband(K, X), wife(K, Y).$$

$$q(X) \leftarrow parent(X, Y).$$
$$q(X) \leftarrow income(X, Y), Y \geq 20000.$$

6.6 For instance:

$$\pi_{X,Y}(Q(Y,X)) \cup \pi_{X,Y}(Q(X,Z) \bowtie R(Z,Y))$$

6.7 For instance:

$$compose(X,Z) \leftarrow r_1(X,Y), r_2(Y,Z).$$

6.9 Take the transitive closure of the *parent*/2-relation.

6.11 For instance:

$$\begin{array}{l}
ingredients(tea, needs(water, needs(tea_bag, nil))).\\
ingredients(boiled_egg, needs(water, needs(egg, nil))).\\
\\
available(water).\\
available(tea_bag).\\
\\
can_cook(X) \leftarrow\\
\qquad ingredients(X, Ingr), all_available(Ingr).\\
\\
all_available(nil).\\
all_available(needs(X,Y)) \leftarrow\\
\qquad available(X), all_available(Y).\\
\\
needs_ingredient(X,Y) \leftarrow\\
\qquad ingredients(X, Ingr), among(Y, Ingr).\\
\\
among(X, needs(X,Y)).\\
among(X, needs(Y,Z)) \leftarrow\\
\qquad among(X,Z).
\end{array}$$

7.1 Alternative list notation:

$$\begin{array}{cc}
.(a, .(b, [\,])) & .(a, .(b, .(c, [\,])))\\
.(a, b) & .(a, .(b, [\,]))\\
.(a, .(.(b, .(c, [\,])), .(d, [\,]))) & .([\,], [\,])\\
.(a, .(b, X)) & .(a, .(b, .(c, [\,])))
\end{array}$$

7.4 For instance:

$$\begin{array}{l}
length([\,], 0).\\
length([X|Y], N) \leftarrow length(Y, M), N \ is \ M + 1.
\end{array}$$

7.5 For instance:

$$lshift([X|YZ], YZX) \leftarrow append(YZ, [X], YZX).$$

7.9 For instance:

$$
\begin{aligned}
&\mathit{sublist}(X, Y) \leftarrow \mathit{prefix}(X, Y)\\
&\mathit{sublist}(X, [Y|Z]) \leftarrow \mathit{sublist}(X, Z).
\end{aligned}
$$

7.12 For instance:

$$
\begin{aligned}
&msort([\,], [\,]).\\
&msort([X], [X]).\\
&msort(X, Y) \leftarrow\\
&\qquad split(X, \mathit{Split1}, \mathit{Split2}),\\
&\qquad msort(\mathit{Split1}, \mathit{Sorted1}),\\
&\qquad msort(\mathit{Split2}, \mathit{Sorted2}),\\
&\qquad merge(\mathit{Sorted1}, \mathit{Sorted2}, Y).
\end{aligned}
$$

$$
\begin{aligned}
&split([X], [], [X]).\\
&split([X, Y|Z], [X|V], [Y|W]) \leftarrow\\
&\qquad split(Z, V, W).
\end{aligned}
$$

$$
\begin{aligned}
&merge([\,], [\,], [\,]).\\
&merge([X|A], [Y|B], [X|C]) \leftarrow\\
&\qquad X < Y, merge(A, [Y|B], C).\\
&merge([X|A], [Y|B], [Y|C]) \leftarrow\\
&\qquad X \geq Y, merge([X|A], B, C).
\end{aligned}
$$

7.13 For instance:

$$
\begin{aligned}
&\mathit{edge}(1, 2, b).\\
&\mathit{edge}(2, 2, a).\\
&\mathit{edge}(2, 3, a).\\
&\mathit{edge}(3, 2, b).
\end{aligned}
$$

$$
\mathit{final}(3).
$$

$$
\begin{aligned}
&\mathit{accept}(\mathit{State}, [\,]) \leftarrow\\
&\qquad \mathit{final}(\mathit{State}).\\
&\mathit{accept}(\mathit{State}, [X|Y]) \leftarrow\\
&\qquad \mathit{edge}(\mathit{State}, \mathit{NewState}, X), \mathit{accept}(\mathit{NewState}, Y).
\end{aligned}
$$

7.17 For instance:

$$
palindrome(X) \leftarrow \mathit{diff\_palin}(X - [\,]).
$$

$$
\begin{aligned}
&\mathit{diff\_palin}(X - X).\\
&\mathit{diff\_palin}([X|Y] - Y).\\
&\mathit{diff\_palin}([X|Y] - Z) \leftarrow \mathit{diff\_palin}(Y - [X|Z]).
\end{aligned}
$$

7.20 Hint: Represent the empty binary tree by the constant *empty* and the nonempty tree by $node(X, Left, Right)$ where X is the label and *Left* and *Right* the two subtrees of the node.

8.3 The following program provides a starting point (the program finds all refutations but it does not terminate):

$$
\begin{array}{l}
prove(Goal) \leftarrow \\
\qquad int(Depth), dfid(Goal, Depth, 0). \\
\\
dfid(true, Depth, Depth). \\
dfid((X, Y), Depth, NewDepth) \leftarrow \\
\qquad dfid(X, Depth, TmpDepth), \\
\qquad dfid(Y, TmpDepth, NewDepth). \\
dfid(X, s(Depth), NewDepth) \leftarrow \\
\qquad clause(X, Y), \\
\qquad dfid(Y, Depth, NewDepth). \\
\\
int(s(0)). \\
int(s(X)) \leftarrow \\
\qquad int(X).
\end{array}
$$

8.4 Hint: For instance, the fourth rule may be defined as follows:

$$d(X + Y, Dx + Dy) \leftarrow d(X, Dx), d(Y, Dy).$$

10.3 Definite clause grammar:

$$
\begin{array}{rcl}
bleat & \rightarrow & [b], aaa. \\
aaa & \rightarrow & [a]. \\
aaa & \rightarrow & [a], aaa.
\end{array}
$$

Prolog program:

$$
\begin{array}{l}
bleat(X_0, X_2) \leftarrow connects(X_0, b, X_1), aaa(X_1, X_2). \\
aaa(X_0, X_1) \leftarrow connects(X_0, a, X_1). \\
aaa(X_0, X_2) \leftarrow connects(X_0, a, X_1), aaa(X_1, X_2). \\
connects([X|Y], X, Y).
\end{array}
$$

A refutation is obtained, for instance, by giving the goal $\leftarrow bleat([b, a, a], [\,])$.

10.4 The DCG describes a language consisting only of the empty string. However, at the same time it defines the “concatenation”-relation among lists. That is, the nonterminal $x([a, b], [c, d], X)$ not only derives the empty string but also binds X to $[a, b, c, d]$.

12.1 The definition of $append/3$ and $member/2$ is left to the reader:

$$
\begin{array}{l}
eq(T1, T2) \leftarrow \\
\qquad nodes(T1, N1), nodes(T2, N2), equal(N1?, N2?). \\
\\
nodes(empty, [\,]). \\
nodes(tree(X, T1, T2), [X|N]) \leftarrow \\
\qquad nodes(T1, N1), nodes(T2, N2), append(N1?, N2?, N). \\
\\
equal(X, Y) \leftarrow \\
\qquad subset(X, Y), subset(Y, X). \\
\\
subset([\,], X). \\
subset([X|Y], Z) \leftarrow \\
\qquad member(X, Z), subset(Y?, Z).
\end{array}
$$

12.2 Hint: write a program which transposes the second matrix and then computes all inner products.

13.3 Hint: The overall structure of the proof is as follows:

$$
\dfrac{\dfrac{E \vdash app(c(X,Y),Z) \doteq c(X, app(Y,Z))}{E \vdash \_ \doteq \_} \qquad \dfrac{E \vdash a \doteq a \qquad \dfrac{E \vdash app(nil, X) \doteq X}{E \vdash \_ \doteq \_}}{E \vdash \_ \doteq \_}}{E \vdash \_ \doteq \_}
$$

14.3 The following program with real-valued or rational constraints can be used to answer e.g. the goal $\leftarrow jugs([M, 1-M, 0], Res, N)$.

$$
\begin{array}{l}
jugs([A, B, C], [A, B, C], N) \leftarrow \\
\qquad N \doteq 0, A + B + C \doteq 1. \\
jugs([A, B, C], Res, N) \leftarrow \\
\qquad N > 0, \\
\qquad jugs([0.6 * A + 0.2 * B, 0.7 * B + 0.4 * A, C + 0.1 * B], Res, N - 1).
\end{array}
$$

15.1 The transformed program looks as follows:

$$
\begin{array}{l}
expr(X, Z) \leftarrow \\
\qquad call\_expr(X, Z), expr(X, [+|Y]), expr(Y, Z). \\
expr([id|Y], Y) \leftarrow \\
\qquad call\_expr([id|Y], Y). \\
call\_expr(X, [+|Y]) \leftarrow \\
\qquad call\_expr(X, Z). \\
call\_expr(Y, Z) \leftarrow \\
\qquad call\_expr(X, Z), expr(X, [+|Y]).
\end{array}
$$

Adding $call\_expr([id, +, id], X)$ to the program yields the semi-naive iteration:

$$
\begin{aligned}
\Delta x_0 &= \{call\_expr([id, +, id], A)\} \\
\Delta x_1 &= \{expr([id, +, id], [+, id]), call\_expr([id, +, id], [+|A])\} \\
\Delta x_2 &= \{call\_expr([id], A), call\_expr([id], [+|A])\} \\
\Delta x_3 &= \{expr([id], [\,])\} \\
\Delta x_4 &= \{expr([id, +, id], [\,])\}
\end{aligned}
$$