Predicate calculus exercises: some answers

Thomas Forster

October 10, 2012

Exercise 1

Render the following fragments of English into predicate calculus, using a lexicon of your choice.

In my model answers I have tended to use bits of English text in `verbatim` *font (as is the habit in certain computer science cultures) for predicate letters, rather than use the single letters that are more customary in most logical cultures. I have done this merely in order to make the notation more suggestive: there is no cultural significance to it. And in any case, further down in the list of model answers I have reverted to the philosophico-logical standard practice of using single capital Roman letters.*

I also write '$\to$' instead of '$\supset$' and '$\wedge$' instead of `&`. *Readers should get used to the idea that there is more than one standard notation for this stuff*

This first bunch involve monadic predicates only and no nested quantifiers.

1. *Every good boy deserves favour; George is a good boy. Therefore George deserves favour.*

   Lexicon.
   Unary predicate letters: `good-boy( )`*;* `deserves-favour( )`
   Constant symbol: `George`

   Formalisation

   $(\forall x)(\texttt{good-boy}(x) \to \texttt{deserves-favour}(x))$*;*
   $\texttt{good-boy}(\texttt{George})$*;*
   therefore $\texttt{deserves-favour}(\texttt{George})$

   You might prefer to have two unary predicate letters `good( )` *and* `boy( )`, *in which case you would have*

   $(\forall x)((\texttt{good}(x) \wedge \texttt{boy}(x)) \to \texttt{deserves-favour}(x))$*;*
   $\texttt{good}(\texttt{George}) \wedge \texttt{boy}(\texttt{George}))$*;*
   therefore $\texttt{deserves-favour}(\texttt{George})$.

2. *All cows eat grass; Daisy eats grass. Therefore Daisy is a cow.*

   Lexicon:
   Unary predicate letters: `eats-grass( )`, `Cow( )`*;*
   Constant symbol: `Daisy`.

   Formalisation

   $(\forall x)(\texttt{Cow}(x) \to \texttt{eats-grass}(x))$
   $\texttt{eats-grass}(\texttt{Daisy})$*;*
   $(\forall x)(\texttt{Cow}(x) \to \texttt{eats-grass}(x))$*;*
   therefore $\texttt{Cow}(\texttt{Daisy})$.

3. *Socrates is a man; all men are mortal. Therefore Socrates is mortal.*

   Lexicon:
   Unary predicate letters: `man( )`, `mortal( )`,
   Constant symbol: `Socrates`.

Formalisation

$\texttt{man}(\texttt{Socrates})$;
$(\forall x)(\texttt{man}(x) \rightarrow \texttt{mortal}(x))$;
therefore $\texttt{mortal}(\texttt{Socrates})$.

4. *Daisy is a cow; all cows eat grass. Therefore Daisy eats grass.*

   Lexicon:
   Unary predicate letters: `eats-grass( )`, `cow( )`;
   Constant symbol: `Daisy`.

   Formalisation

   $\texttt{cow}(\texttt{Daisy})$;
   $(\forall x)(\texttt{cow}(x) \rightarrow \texttt{eats-grass}(x))$;
   therefore $\texttt{eats-grass}(\texttt{Daisy})$.

5. *Daisy is a cow; all cows are mad. Therefore Daisy is mad.*

   Lexicon:
   Unary predicate letters: `mad( )`, `cow( )`;
   Constant symbol: (`Daisy`).

   Formalisation

   $\texttt{cow}(\texttt{Daisy})$;
   $(\forall x)(\texttt{cow}(x) \rightarrow \texttt{mad}(x))$;
   therefore $\texttt{mad}(\texttt{Daisy})$.

6. *No thieves are honest; some dishonest people are found out. Therefore some thieves are found out.*

   Lexicon:
   Unary predicate letters: `thief( )`, `honest( )`, `found-out( )`.

   Formalisation

   $(\forall x)(\texttt{thief}(x) \rightarrow \neg(\texttt{honest}(x))$;
   $(\exists x)(\neg\texttt{honest}(x) \wedge \texttt{found-out}(x))$;
   therefore $(\exists x)(\texttt{thief}(x) \wedge \texttt{found-out}(x))$.

7. *No muffins are wholesome; all puffy food is unwholesome. Therefore all muffins are puffy.*

   Lexicon:
   Unary predicate letters: `muffin( )`, `wholesome( )`, `puffy( )`.

   Formalisation

   $\neg(\exists x)(\texttt{muffin}(x) \wedge \texttt{wholesome}(x))$;
   $(\forall x)(\texttt{puffy}(x) \rightarrow \neg(\texttt{wholesome}(x)))$;
   therefore $(\forall x)(\texttt{muffin}(x) \rightarrow \texttt{puffy}(x))$.

8. *No birds except peacocks are proud of their tails; some birds that are proud of their tails cannot sing. Therefore some peacocks cannot sing.*

   Lexicon:
   Unary predicate letters: $\texttt{peacock( )}$, $\texttt{can-sing( )}$, $\texttt{proud-of-tail( )}$.

   Formalisation

   $(\forall x)(\texttt{proud-of-tail}(x) \to \texttt{peacock}(x))$;
   $(\exists x)(\texttt{proud-of-tail}(x) \wedge \neg(\texttt{can-sing}(x)))$;
   therefore $(\exists x)(\texttt{peacock}(x) \wedge \neg(\texttt{can-sing}(x)))$.

9. *A wise man walks on his feet; an unwise man on his hands. Therefore no man walks on both.*

   Formalisation

   $(\forall x)(\texttt{wise}(x) \to \texttt{walks-on-feet}(x))$;
   $(\forall x)(\neg(\texttt{wise}(x)) \to \texttt{walks-on-hands}(x))$;
   $(\forall x)(\neg(\texttt{walks-on-feet}(x) \wedge \texttt{walks-on-hands}(x)))$.

   You might want to try to capture the fact that $\texttt{walks-on-feet( )}$ *and* $\texttt{walks-on-hands( )}$ *share some structure, and accordingly have a two-place relation* $\texttt{walks-on(,)}$. *Then i think you will also want binary predicate letters* $\texttt{feet-of(,)}$ *and* $\texttt{hands-of(,)}$ *so you would end up with*

   $(\forall x)(\texttt{wise}(x) \to (\forall y)(\texttt{feet-of}(x, y) \to \texttt{walks-on}(x, y)))$

   and of course

   $(\forall x)(\neg\texttt{wise}(x) \to (\forall y)(\texttt{hands-of}(x, y) \to \texttt{walks-on}(x, y)))$

   You might feel that the following are equally good formalisations:

   $(\forall x)(\texttt{wise}(x) \to (\exists y)(\exists z)(\texttt{feet-of}(x, y) \wedge \texttt{feet-of}(x, z) \wedge \neg(y = z) \wedge \texttt{walks-on}(x, y) \wedge \texttt{walks-on}(x, z)))$ *... and the same for unwise men and hands.*

   However that involves **two***-place relations and we haven't got to them yet!*

10. *No fossil can be crossed in love; an oyster may be crossed in love. Therefore oysters are not fossils.*

    Lexicon:
    Unary predicate letters: $\texttt{fossil( )}$. $\texttt{oyster( )}$, $\texttt{crossed-in-love( )}$.

    Formalisation

    $(\forall x)(\texttt{fossil}(x) \to \neg\texttt{can-be-crossed-in-love}(x))$;
    $(\forall x)(\texttt{oyster}(x) \to \texttt{can-be-crossed-in-love}(x))$;
    therefore $(\forall x)(\texttt{oyster}(x) \to \neg\texttt{fossil}(x))$

11. *All who are anxious to learn work hard; some of these students work hard. Therefore some of these students are anxious to learn.*

Lexicon:
Unary predicate letters: `anxious-to-learn( )`, `works-hard( )`, `student( )`.

Formalisation

$(\forall x)(\texttt{anxious-to-learn}(x) \to \texttt{works-hard}(x))$;
$(\exists x)(\texttt{student}(x) \wedge \texttt{works-hard}(x))$;
therefore $(\exists x)(\texttt{student}(x) \wedge \texttt{anxious-to-learn}(x))$.

12. *His songs never last an hour. A song that lasts an hour is tedious. Therefore his songs are never tedious.*

Lexicon:
Unary predicate letters: `last-an-hour( )`, `song( )`, `his( )`, `tedious( )`.

Formalisation

$(\forall y)((\texttt{song(y)} \wedge (\texttt{his}(y)) \to \texttt{last-an-hour}(y))$;
$(\forall x)((\texttt{song}(x) \wedge \texttt{last-an-hour}(x)) \to \texttt{tedious}(x))$;
therefore $(\forall z)((\texttt{song}(z) \wedge \texttt{his}(z)) \to \neg\texttt{tedious}(z))$.

13. *Some lessons are difficult; what is difficult needs attention. Therefore some lessons need attention.*

Lexicon.
Unary predicate letters: `lesson( )`, `difficult( )`, `needs-attention( )`.

Formalisation

$(\exists x)(\texttt{lesson}(x) \;\wedge\; \texttt{difficult}(x))$;
$(\forall z)(\texttt{difficult}(z) \to \texttt{needs-attention}(z))$.
therefore $(\exists x)(\texttt{lesson}(x) \wedge \texttt{needs-attention}(x))$.

14. *All humans are mammals; all mammals are warm blooded. Therefore all humans are warm-blooded.*

Lexicon.
Unary predicate letters: `human( )`, `mammal( )`, `warm-blooded( )`.

Formalisation

$(\forall y)(\texttt{human}(y) \to \texttt{mammal}(y))$;
$(\forall y)(\texttt{mammal}(y) \to \texttt{warmblooded}(y))$;
therefore $(\forall z)(\texttt{human}(z) \to \texttt{warmblooded}(z))$.

15. *Warmth relieves pain; nothing that does not relieve pain is useful in toothache. Therefore warmth is useful in toothache.*

Lexicon:
Unary predicate letters: `relieves-pain( )`, `useful-in-toothache( )`;
Constant symbol: `warmth`,

Formalisation

> $\texttt{relieves-pain}(\texttt{warmth})$;
> $(\forall x)(\texttt{useful-in-toothache}(x) \to \texttt{relieves-pain}(x)$;
> *therefore* $\texttt{useful-in-toothache}(\texttt{warmth})$

You might want to break up `relieves-pain` *by having a binary predicate letter* `relieves( , )` *and a constant symbol* `pain`*, giving*

$\texttt{relieves}(\texttt{warmth}, \texttt{pain})$;
$(\forall x)(\texttt{useful-in-toothache}(x) \to \texttt{relieves}(x, \texttt{pain})$;
therefore $\texttt{useful-in-toothache}(\texttt{warmth})$.

16. *Guilty people are reluctant to answer questions;*

17. *Louis is the King of France; all Kings of France are bald. Therefore Louis is bald.*

    Lexicon:
    Unary predicate letters: `bald( )`, `King-of-France( )`,
    Constant symbol: `Louis`.

    Formalisation

    > $\texttt{king-of-France}(\texttt{Louis})$;
    > $(\forall x)(\texttt{king-of-France}(x) \to \texttt{bald}(x))$;
    > *therefore* $\texttt{bald}(\texttt{Louis})$.

    You might feel that `King-of-France` *is not really a unary predicate but a binary predicate* (`king-of`) *with one argument place plugged by a constant* (`France`).

Exercise 2 *Render the following into Predicate calculus, using a lexicon of your choice. These involve nestings of more than one quantifier, polyadic predicate letters, equality and even function letters.*

1. *Anyone who has forgiven at least one person is a saint.*

   Lexicon:
   Unary predicate letters: `saint( )`
   Binary predicate letters: `has-forgiven( , )`

   Formalisation

   > $(\forall x)(\forall y)(\texttt{has-forgiven}(x, y) \to \texttt{saint}(x))$

2. *Nobody in the logic class is cleverer than everybody in the history class.*

   Lexicon:
   Unary predicate letters: `is-in-the-logic-class( )`, `is-in-the-history-class( )`
   Binary predicate letter: `is-cleverer-than( , )`

   Formalisation

   > $(\forall x)(\texttt{is-in-the-logic-class}(x) \to (\exists y)(\texttt{is-in-the-history-class}(y) \wedge$
   > $\neg(\texttt{is-cleverer-than}(x, y))$;

Here you might prefer to have a two-place relation between people and subjects, so that you then have two constants, `history` *and* `logic`.

3. *Everyone likes Mary—except Mary herself.*

   Lexicon:
   Binary predicate letter: $L(\ ,\ )$
   Constant symbol: m

   Formalisation

   $$\neg L(m, m) \wedge (\forall x)(x \neq m \rightarrow L(x, m))$$

4. *Jane saw a bear, and Roger saw one too.*

   Lexicon:
   Unary predicate letter: $B(\ )$
   Binary predicate letter: $S(\ ,\ )$
   Constant symbols: j, r

   Formalisation

   $$(\exists x)(B(x) \wedge S(j, x)) \wedge (\exists x)(B(x) \wedge S(r, x));$$

5. *Jane saw a bear and Roger saw it too.*

   $$(\exists x)(B(x) \wedge S(j, x) \wedge S(r, x))$$

6. *God will destroy the city unless there is a righteous man in it;*

7. *Some students are not taught by every teacher;*

   Lexicon:
   Unary predicate letters: `teacher`( ), `student`( ).
   Binary predicate letter: `taught-by`(,)

   Formalisation

   $$(\exists x)\texttt{student}(x) \wedge \neg(\forall y)(\texttt{teacher}(y) \rightarrow \texttt{taught-by}(x, y))$$

   Of course you might want to replace '$\texttt{teacher}(x)$*' by '*$(\exists y)(\texttt{taught-by}(y, x)$*'.*

8. *No student has the same teacher for every subject.*

   Lexicon:
   Ternary predicate letter: $R(\ ,\ ,\ )$
   Unary predicate letters: `student`( ), `teacher`( ), `subject`( ).

   Formalisation

   $$(\forall x)(\texttt{student}(x) \rightarrow \neg(\forall y)(\texttt{teacher}(y) \rightarrow (\forall z)(\texttt{subject}(z) \rightarrow R(x, y, z))))$$

9. *Everybody loves my baby, but my baby loves nobody but me.*

   Lexicon:
   Binary predicate letter[1]*:* $L(\ ,\ )$;
   Constant symbols: b, m.

   Formalisation

   $$(\forall x)(L(x,b)) \wedge (\forall x)(L(b,x) \rightarrow x = m);$$

Exercise 3 *Match up the formulæ on the left with their English equivalents on the right.*

(i) $(\forall x)(\exists y)(x \text{ loves } y)$	*(a)* *Everyone loves someone*
(ii) $(\forall y)(\exists x)(x \text{ loves } y)$	*(b)* *There is someone everyone loves*
(iii) $(\exists y)(\forall x)(x \text{ loves } y)$	*(c)* *There is someone that loves everyone*
(iv) $(\exists x)(\forall y)(x \text{ loves } y)$	*(d)* *Everyone is loved by someone*

These involve nested quantifiers and dyadic predicates

Exercise 4 *Render the following pieces of English into Predicate calculus, using a lexicon of your choice.*

1. *Everone who loves is loved;*

   $$(\forall x)(\forall y)(L(y,x) \rightarrow (\exists z)(L(z,y))$$

2. *Everyone loves a lover;*

   $$(\forall x)(\forall y)(L(y,x) \rightarrow (\forall z)(L(z,y))$$

3. *The enemy of an enemy is a friend*

4. *The friend of an enemy is an enemy*

5. *Any friend of George's is a friend of mine*

6. *Jack and Jill have at least two friends in common*

7. *Two people who love the same person do not love each other.*

8. *None but the brave deserve the fair.*

   $$(\forall x)(\forall y)((F(x) \wedge D(y,x)) \rightarrow B(y))$$

9. *If there is anyone in the residences with measles then anyone who has a friend in the residences will need a measles jab.*

[1] Observe that we do not have to specify that '=' is part of the lexicon. That's a given, since it is part of the logical vocabulary.

This next batch involves nested quantifiers and dyadic predicates and equality

Exercise 5 *Render the following pieces of English into Predicate calculus, using a lexicon of your choice.*

1. *There are two islands in New Zealand;*

2. *There are* three[2] *islands in New Zealand;*

3. *tf knows (at least) two pop stars;*

   (You must resist the temptation to express this as a relation between tf and a plural object consisting of two pop stars coalesced into a kind of plural object like Jeff Goldblum and the Fly. You will need to use '=', the symbol for equality.)

$$(\exists xy)(x \neq y \wedge K(x) \wedge K(y))$$

   '$K(x)$' of course means that x is a pop star known to me.

4. *You are loved only if you yourself love someone [other than yourself!];*

$$(\forall x)(\forall y)(L(y,x) \to (\exists z)(z \neq x \wedge L(x,z))$$

5. *The only living Nobel prizewinner I know is Andrew Huxley.*

6. *God will destroy the city unless there are (at least) two righteous men in it;*

7. *There is at most one king of France;*

$$(\forall xy)(K(x) \wedge K(y) \to x = y)$$

8. *I know no more than two pop stars;*

$$(\forall xyz)((K(x) \wedge K(y) \wedge K(z)) \to (x = y \vee x = z \vee y = z))$$

9. *There is precisely one king of France;*

$$(\exists x)(K(x) \wedge (\forall y)(K(y) \to y = x))$$

   Notice that

$$(\exists x)(\forall y)(K(x) \wedge (K(y) \to y = x))$$

   would do equally well. Make sure you are happy about this.

10. *I know three FRS's and one of them is bald;*

[2] The third is Stewart Island

11. *Brothers and sisters have I none; this man's father is my father's son.*
12. ** Anyone who is between a rock and a hard place is also between a hard place and a rock.*