CHAPTER 8

Predicates and quantifiers

8.1 Introduction

A *predicate* is a logical statement that depends on a value or values. When a predicate is applied to a particular value it becomes a *proposition*. An example is the predicate:

prime(x)

which depends on some numeric value, x, so that

prime(7)

is a true proposition meaning that seven is a prime number, and

prime(6)

is a false proposition. (A prime number is one that is only divisible by itself and 1).

8.2 Quantifiers

Quantifiers can be applied to predicates to give propositions.

8.2.1 Universal quantifier

The *universal quantifier* is written:

$\forall$

and is pronounced 'for all' (it looks like an upside-down 'A', in *for All*). It is used in the form:

$\forall$ declaration | constraint • predicate

which states that for the declaration(s) given, restricted to certain values (by a predicate called a constraint), the predicate holds.

For each of the quantifiers to be described here, the

| constraint

part may be omitted.

The declaration introduces a typical element that is then optionally constrained. The predicate may apply to this element. For example, to state that all natural numbers less than 10 have squares less than 100:

$$\forall\, i : \mathbb{N} \mid i < 10 \bullet i^2 < 100$$

This would be pronounced: 'for all *i* drawn from the set of natural numbers, such that *i* is under ten, *i* squared is less than 100'.

A universal quantification can be thought of as a chain of conjunctions. The quantification above is equivalent to:

$$0^2 < 100 \land 1^2 < 100 \land \ldots \land 8^2 < 100 \land 9^2 < 100$$

If the set of values over which the variable is universally quantified is *empty*, then the quantification is defined to be true:

$$\forall\, i : \mathbb{N} \mid 0 \leq i < 0 \bullet i^2 < 100 \quad \text{is defined to be true}$$

8.2.2 Existential quantifier

The *existential quantifier* is written:

$$\exists$$

and is pronounced 'there exists' (it looks like a backwards 'E', in *there Exists*). It is used in the form:

$$\exists \text{ declaration} \mid \text{constraint} \bullet \text{predicate}$$

The declaration introduces a typical variable which is then optionally constrained. The predicate applies to this variable. For example, to state that there is a natural number under ten which has a square less than 100:

$$\exists\, i : \mathbb{N} \mid i < 10 \bullet i^2 < 100$$

This would be pronounced: 'there exists an *i* drawn from the set of natural numbers, such that *i* is less than ten, and *i* squared is less than 100'.

Note that there need not be only one value of *i* for which this is true; in this example there are ten: 0 to 9.

To define the predicate *Even*:

$$Even(x) \Leftrightarrow \exists\, k : \mathbb{Z} \bullet k * 2 = x$$

existential

A universal quantification can be thought of as a chain of disjunctions.

If the set of values over which the variable is quantified is *empty*, then the existential quantification is defined to be false:

$$\exists\, i : \mathbb{N} \mid 0 \leq i < 0 \bullet i^2 < 100 \quad \text{is defined to be false}$$

8.2.3 Unique quantifier

The *unique quantifier* is similar to the existential quantifier except that it states that there exists *only one* value for which the predicate is true.

The unique quantifier is written:

$$\exists_1$$

An example is:

$$\exists_1\ i : \mathbb{N} \mid i < 10 \bullet i^2 < 100 \land i^2 > 80$$

This would be pronounced: 'there exists only one *i* drawn from the set of natural numbers, where *i* is less than ten, such that *i* squared is less than 100 and *i* squared is greater than 80'. It is equivalent to saying that the predicate holds for *i*, but that there is no *j* (with a value different from *i*) for which it holds:

$$\begin{array}{l} \exists_1\ i : \mathbb{N} \mid i < 10 \bullet i^2 < 100 \land i^2 > 80 \\ \Leftrightarrow \\ \exists\ i : \mathbb{N} \mid i < 10 \bullet i^2 < 100 \land i^2 > 80 \\ \land \neg(\exists\ j : \mathbb{N} \mid j < 10 \land i \neq j \bullet j^2 < 100 \land j^2 > 80) \end{array}$$

8.2.4 Counting quantifier

Some notations use a counting quantifier that counts for how many values of the variable the predicate holds. In Z this is not needed; instead, we use a set comprehension to construct the set of values for which the predicate holds, and then finds the size of the set.

An example is: the number of natural numbers under 10 that have squares greater than 30:

$$\#\{\, i : \mathbb{N} \mid i < 10 \bullet i^2 > 30 \,\}$$

8.2.5 Quantifiers in schema

Quantifiers may be used in the expressions contained in the predicate part of a schema.

8.3 Set comprehension

So far we have given values to sets by listing all their elements. It is also possible to give a value to a set by giving a condition (a *predicate*) which must hold for all members of the set. This can be done by a formulism called a *set comprehension*. The general form is:

$$\{\text{declaration} \mid \text{constraint} \bullet \text{expression}\}$$

- The *declaration* is for a typical element and it gives the element's type.
- The *constraint* restricts the possible values of the typical element. It is a logical expression which must be true for that value of the typical element to be included.
- The *expression* is an expression indicating the value to be included in the set.

A comprehension is very useful for giving a value to an infinite set. For example we cannot write:

$\{\ldots -8, -6, -4, -2, 0, 2, 4, 6, 8, \ldots\}$

since we would then rely on solely the reader's intuition to understand what the continuation indicated by '...' must be. Instead we write:

$\{x: \mathbb{Z} \mid Even(x) \bullet x\}$

Here *x* is the typical value. It is of type *integer* so the set generated will be a *set* of integers. The value of *x* is constrained to be even. The value *x* is included in the generated set. So the generated set is the set of even integers.

The following set comprehension generates the set of the *squares* of the *even* integers:

$\{x: \mathbb{Z} \mid Even(x) \bullet x * x\}$

The constraint and its preceding bar may be omitted:

$\{x: \mathbb{N} \bullet x * x\}$ the squares of the natural numbers

8.3.1 Ranges of numbers

The notation

$m \mathrel{..} n$

was introduced in Chapter 2. It is shorthand for

$\{ i: \mathbb{Z} \mid m \leq i \land i \leq n \bullet i \}$

8.4 Relationship between logic and set theory

There is a direct relationship between some of the operators of logic and operations on sets:

$[X]$ any set
$S, T: \mathbb{P}X$
$S \cup T == \{x: X \mid x \in S \lor x \in T \bullet x\}$
$S \cap T == \{x: X \mid x \in S \land x \in T \bullet x\}$
$S \setminus T == \{x: X \mid x \in S \land x \notin T \bullet x\}$

8.5 Summary of notation

$\forall\ x{:}T \bullet P$	Universal quantification: 'for all x of type T, P holds'
$\exists\ x{:}T \bullet P$	Existential quantification: 'there exists an x of type T, such that P holds'
$\exists_1\ x{:}T \bullet P$	Unique existence: 'there exists a unique x of type T, such that P holds'
$\{D \mid P \bullet t\}$	the set of t's declared by D where P holds

EXERCISES

1. Re-express the proposition

   $\forall p{:}\ PERSON \mid p \in loggedIn \bullet p \in users$

   using set relations.
2. Write an expression that states that the squares of all integers are non-negative.
3. Write an expression to state that there is a number that is equal to itself squared.
4. Using the fact that *m mod n* is zero when *m* is divisible by *n* ($m \geq 0$ and $n > 0$) write a set comprehension to define the set of prime numbers.