```
print "Basset hounds got long ears" if length $ear >= 10;
go_outside() and play() unless $is_raining;
```

The foreach modifier is an iterator: it executes the statement once for each item in the LIST (with $_ aliased to each item in turn).

```
print "Hello $_!\n" foreach qw(world Dolly nurse);
```

while repeats the statement *while* the condition is true. until does the opposite, it repeats the statement *until* the condition is true (or while the condition is false):

```
# Both of these count from 0 to 10.
print $i++ while $i <= 10;
print $j++ until $j >  10;
```

The while and until modifiers have the usual "while loop" semantics (conditional evaluated first), except when applied to a do-BLOCK (or to the deprecated do-SUBROUTINE statement), in which case the block executes once before the conditional is evaluated. This is so that you can write loops like:

```
do {
    $line = <STDIN>;
    ...
} until $line  eq ".\n";
```

See do in *perlfunc*. Note also that the loop control statements described later will *NOT* work in this construct, because modifiers don't take loop labels. Sorry. You can always put another block inside of it (for next) or around it (for last) to do that sort of thing. For next, just double the braces:

```
do {{
    next if $x == $y;
    # do something here
}} until $x++ > $z;
```

For last, you have to be more elaborate:

```
LOOP: {
        do {
            last if $x = $y**2;
            # do something here
        } while $x++ <= $z;
}
```

NOTE: The behaviour of a my statement modified with a statement modifier conditional or loop construct (e.g. my $x if ...) is **undefined**. The value of the my variable may be undef, any previously assigned value, or possibly anything else. Don't rely on it. Future versions of perl might do something different from the version of perl you try it out on. Here be dragons.

26.1.6 Compound Statements

In Perl, a sequence of statements that defines a scope is called a block. Sometimes a block is delimited by the file containing it (in the case of a required file, or the program as a whole), and sometimes a block is delimited by the extent of a string (in the case of an eval).

But generally, a block is delimited by curly brackets, also known as braces. We will call this syntactic construct a BLOCK.

The following compound statements may be used to control flow: