5 Flow of control: branches

We have seen the word `if` used to control which instruction is executed next. Commonly, we want to do one thing in one case and another thing in a different case. An `if` can be followed by an `elsif` (or more than one `elsif`), with an `else` at the end to catch any remaining possibilities:

(4)

```
1  if ($price >= 100)
2    {
3    print "It\'s expensive.\n";
4    }
5  elsif ($price > 0)
6    {
7    print "It\'s cheap.\n";
8    }
9  elsif ($price == 0)
10   {
11   print "It\'s free.\n";
12   }
13 else
14   {
15   print "Cost is negative.\n";
16   print "That can\'t be right!\n";
17   }
```

When any one of the tests is passed, the remaining tests are ignored; if `$price` is 200, then since 200 ≥ 100 Perl will print `It's expensive`, and the message in 4.7 will not be printed even though it is also true that 200 > 0.

Curly brackets are used to keep together the *block* of code to be executed if a test is passed. Notice that (unlike in some programming languages) even if the block contains just a single line of code, that line must still have curly brackets round it. The last statement before the `}` does not actually have to end in a semicolon, but it is sensible to include one anyway. We might want to modify our code by adding further statements, in which case it would be easy to overlook the need to add a missing semicolon.