```
flag = false;
...
flag = true;
```

Because the `<stdbool.h>` header is so handy, I'll use it in subsequent programs whenever Boolean variables are needed.

5.3 The `switch` Statement

In everyday programming, we'll often need to compare an expression against a series of values to see which one it currently matches. We saw in Section 5.2 that a cascaded `if` statement can be used for this purpose. For example, the following cascaded `if` statement prints the English word that corresponds to a numerical grade:

```
if (grade == 4)
  printf("Excellent");
else if (grade == 3)
  printf("Good");
else if (grade == 2)
  printf("Average");
else if (grade == 1)
  printf("Poor");
else if (grade == 0)
  printf("Failing");
else
  printf("Illegal grade");
```

As an alternative to this kind of cascaded `if` statement, C provides the `switch` statement. The following `switch` is equivalent to our cascaded `if`:

```
switch (grade) {
  case 4:  printf("Excellent");
           break;
  case 3:  printf("Good");
           break;
  case 2:  printf("Average");
           break;
  case 1:  printf("Poor");
           break;
  case 0:  printf("Failing");
           break;
  default: printf("Illegal grade");
           break;
}
```

When this statement is executed, the value of the variable `grade` is tested against 4, 3, 2, 1, and 0. If it matches 4, for example, the message `Excellent` is printed, then the `break` statement transfers control to the statement following the `switch`. If the value of `grade` doesn't match any of the choices listed, the `default` case applies, and the message `Illegal grade` is printed.

`break` statement ➤6.4