Worksheet 4 - *If, else and case statements*

Within almost all programming languages, the *if statement* is one of the most important statements that you will use. The *if statement* allows the selective execution of blocks of code. The syntax is as follows:

```
if <expression> then begin
        <code to run if expression is true>
end
else begin
        <code to run if expression is false>
end;
```

It is not necessary to include the *else* part of the statement if you don't want anything to happen when the expression is false (note the lack of the semicolon before the else, however a semicolon is needed if the else is not present). The expression is generally of the form '<variable1> <comparison operator> <variable2>' (e.g. temp > 3) but can take the form <variable1> (e.g. *if temp* is the same as *if temp = true*). The valid comparison operators are:

=	Tests if two variables are exactly equal.
>, <	Tests if the variables are greater/lesser than each other.
>=, <=	Tests if the variables are greater/lesser or equal to each other.
<>	Tests if the variables are different.

The comparison operators produce a boolean result; this can be combined with the boolean operators to test multiple things at once. For example the expression could be '(a = b) **and** (c >= d)'. It is also possible to use the other operators (for the types being compared) on the variables before they are compared. As a result it is possible to have expressions such as '(a + b) <= c'.

In Delphi you can also 'nest' if statements within one another to test for multiple things, for example:

```
if (a = b) then begin
  if (c > d) then begin <code> end
  else begin <code> end;
end
else if (a = c) then begin <code>
end;
```

An alternative to having numerous else if statements is to use the case statement. Its syntax is as follows:

```
case X of
  X1: begin <code to do if X = X1> end;
  X2: begin <code to do if X = X2> end;
  else begin <code to do if all else fails> end;
end;
```

Unfortunately when using the case statement, X must be an integer or a character and X1, X2, … must be constants. However, when circumstances permit, it can save a considerable amount of code.

- Write two expressions representing the following conditions using: a single nested if statement, a series of if statements using the 'and' operator.
  - B is greater than C and B is greater than D then do 1.
  - B is equal to C and B is greater than D then do 2.
  - B is equal to C and B is equal to D then do 3.
  - B is greater than C and B isn't greater than D then do 4.
  - B is equal to C and B less than D then do 5.