4 The Boolean and CHAR Types

4.1 Boolean Constants and Operations

Many things can be expressed in terms of two mutually exclusive states. For example, either I am at home or I am not; either I have enough money to buy the latest Porsche or I have not. It is therefore important to be able to define variables which can take precisely two values. Such variables are called logical variables or boolean variables (after the mathematician George Boole). In MODULA-2 there is a predefined type BOOLEAN. Any variable of type BOOLEAN can take one of the two values FALSE or TRUE, where FALSE and TRUE are standard identifiers. The two examples above can thus be expressed as follows:

```
VAR
    AtHome, BuyPorsche: BOOLEAN;
    ;
    AtHome: = FALSE; (* I am not at home *)
    AtHome: = TRUE; (* I am at home *)
    BuyPorsche: = FALSE; (* I cannot buy a Porsche *)
    BuyPorsche: = TRUE; (* I can buy a Porsche *)
```

There are three operators which can be applied to boolean values, each giving a boolean value as result:

1) The AND Operator
   When this operator is applied to two boolean values P and Q, the result is TRUE only if both P and Q are TRUE, otherwise it is FALSE. The symbol & may be used instead of AND.

2) The OR Operator
   This operator is also applied to two boolean values P and Q. The result is FALSE only if both P and Q are FALSE, otherwise it is TRUE.

3) The NOT Operator
   This operator is applied to a single boolean value P. The result is TRUE if P is FALSE and FALSE if P is TRUE.

Figure 4.1 shows the effects of the different operators.

4.2 Boolean Expressions

More than one boolean operator may be involved in a boolean expression.

   e.g. (P AND Q) OR NOT (R AND S)
### simple expression

![Diagram of simple expression](image)

### term

![Diagram of term](image)

### factor

![Diagram of factor](image)

---

**Fig. 4.1** The boolean operators

<table>
<thead>
<tr>
<th>P</th>
<th>Q</th>
<th>P and Q</th>
<th>P OR Q</th>
<th>NOT P</th>
</tr>
</thead>
<tbody>
<tr>
<td>FALSE</td>
<td>FALSE</td>
<td>FALSE</td>
<td>FALSE</td>
<td>TRUE</td>
</tr>
<tr>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>TRUE</td>
</tr>
<tr>
<td>TRUE</td>
<td>FALSE</td>
<td>FALSE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
<tr>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>TRUE</td>
<td>FALSE</td>
</tr>
</tbody>
</table>

---

**Fig. 4.2** Syntactic diagrams of a simple expression