BOOLEAN LOGIC OPERATORS

The **IF-THEN-ELSE-ENDIF**, **WHILE-WEND**, and **REPEAT-UNTIL** conditions now support the logical operators **NOT**, **AND**, **OR**, and **XOR**. The **NOT** operator inverts the outcome of a condition, changing false to true, and true to false. The following two **IF-THEN** conditions are equivalent: -

```
IF VAR1 <> 100 THEN NotEqual          ' Goto notEqual if VAR1 is not 100.
IF NOT VAR1 = 100 THEN NotEqual       ' Goto notEqual if VAR1 is not 100.
```

The operators **AND**, **OR**, and **XOR** join the results of two conditions to produce a single true/false result. **AND** and **OR** work the same as they do in everyday speech. Run the example below once with **AND** (as shown) and again, substituting **OR** for **AND**: -

```
DIM VAR1 AS BYTE
DIM VAR2 AS BYTE

CLS
VAR1 = 5
VAR2 = 9
IF VAR1 = 5 AND VAR2 = 10 THEN Res_True
STOP

Res_True:
    PRINT "RESULT IS TRUE."
    STOP
```

The condition "VAR1 = 5 **AND** VAR2 = 10" is not true. Although VAR1 is 5, VAR2 is not 10. **AND** works just as it does in plain English, both conditions must be true for the statement to be true. **OR** also works in a familiar way; if one or the other or both conditions are true, then the statement is true. **XOR** (short for exclusive-OR) may not be familiar, but it does have an English counterpart: If one condition or the other (but not both) is true, then the statement is true.

Parenthesis (or rather the lack of it!).

Every compiler has it's quirky rules, and the PROTON+ compiler is no exception. One of its quirks means that parenthesis is not supported in a Boolean condition, or indeed with any of the **IF-THEN-ELSE-ENDIF**, **WHILE-WEND**, and **REPEAT-UNTIL** conditions. Parenthesis in an expression within a condition is allowed however. So, for example, the expression: -

```
IF (VAR1 + 3) = 10 THEN do something.          Is allowed.
```

But: -

IF((VAR1 + 3) = 10) **THEN** do something. **Is NOT allowed.**

The Boolean operands do have a precedence in a condition. The **AND** operand has the highest priority, then the **OR**, then the **XOR**.

This means that a condition such as: -

IF VAR1 = 2 **AND** VAR2 = 3 **OR** VAR3 = 4 **THEN** do something

Will compare VAR1 and VAR2 to see if the **AND** condition is true. It will then see if the **OR** condition is true, based on the result of the **AND** condition.

THEN operand always required.

The PROTON+ compiler relies heavily on the **THEN** part. Therefore, if the **THEN** part of a condition is left out of the code listing, a SYNTAX ERROR will be produced.