2. Logical Operations

Using logical operators: NOT, AND, OR, and XOR

Logical operators are used in programming. It is not unusual to find a code such as:

- IF A = 1 **AND** B = 1 THEN
- IF A = 1 **OR** B = 1 THEN

The rules for combining logical expressions are usually written down as tables that list all the possible outcomes. These tables are known as truth tables.

NOT

The NOT logical operator has only one input and one output. The output is the opposite of the input.

Input (A)	Output (NOT A)
0	**1**
1	**0**

AND

The AND logical operator has two inputs and one output. The output is 1 only if A and B are both 1.

Input (A)	Input (B)	Output (A AND B)
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

OR

The OR logical operator has two inputs and one output. The output is 1 if either A or B is 1.

Input (A)	Input (B)	Output (A OR B)
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

XOR

The XOR logical operator has two inputs and one output. The output is 1 only if A and B are different.

Input (A)	Input (B)	Output (A XOR B)
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

Logical operations

Logical operations can be used in control systems. For example, a control system that is required to close the windows on a commercial greenhouse when at least one of the following conditions is true:

- the wind speed rises above 12 km per hour.
- it is raining.

would use the logical operator **OR**.

A control system that is required to turn on a sprinkler system in a field when both of the following conditions are true:

- the temperature rises above 25° Celsius
- it has not rained in the last five days

would use the logical operator **AND**.

Boolean algebra

In addition to truth tables, Boolean algebra can be used to describe logical expressions. Logical expressions can become complex and Boolean algebra is a method that can be used to simplify these expressions.

Notation

The logical operators that are going to be used are NOT, AND, OR and XOR. Boolean algebra uses the following notation to represent the input variables and operators:

- a bar on top of and input variable represents the NOT operator. NOT A = $\bar{A}$
- a dot "." represents the AND function. A and B = A.B
- a plus sign "+" represents the OR function. A or B = A + B
- a plus sign with a circle represents the XOR function. A exclusive or B = $A \oplus B$

Laws of Boolean Algebra

Boolean Algebra is a system of mathematics based on logic that has its own set of rules or laws that can be used to simplify Boolean expressions.

<u>Annulment Law</u>

A.0 = 0 A variable AND 0 is always equal to 0

A + 1 = 1 A variable OR 1 is always equal to 1

Identity Law

A + 0 = A A variable OR 0 is always equal to the variable

A.1 = A A variable AND 1 is always equal to the variable

Idempotent Law

A + A = A A variable OR itself is always equal to the variable

A.A = A A variable AND itself is always equal to the variable

Complement Law

A.$\bar{A}$ = 0 A variable AND its complement is always equal to 0

A + $\bar{A}$ = 1 A variable OR its complement is always equal to 1

Commutative Law

A.B = B.A The order of two variables with AND makes no difference

A + B = B + A The order of two variables with OR makes no difference

Double Complement Law

$\bar{\bar{A}}$ = A A double complement of a variable is always equal to the variable

Distributive Law

A(B + C) = A.B + A.C (OR Distributive law)

A + (B.C) = (A + B).(A + C) (AND Distributive law)

Absorptive law

A + (A.B) = A (OR Absorption law)

A(A + B) = A (AND Absorption law)

Associative Law

A + (B + C) = (A + B) + C = A + B + C (OR Associative law)

A(B.C) = (A.B)C = A.B.C (AND Associative law)

Examples - using the rules.

1. Simplify the Boolean expression:

$$X = A.B + A.\bar{B}$$
$$X = A.(B + \bar{B}) \quad \text{Distributive law}$$
$$X = A.1 \quad \text{Inverse law}$$
$$X = A \quad \text{Identity law}$$

2. Prove that $A + \bar{A}.B = A + B$

$A + \bar{A}.B = A.1 + \bar{A}.B$	Identity law
$= A.(1 + B) + \bar{A}.B$	Annulment law
$= A + A.B + \bar{A}.B$	Distributive law
$= A + B.(A + \bar{A})$	Inverse law
$= A + B$	