### Boolean Logic

Computer errors or "bugs" are the result of flaws in the thinking or logic of humans.  If a computer is programmed correctly it should be able to apply simple rules of logic to a given problem and solve the problem correctly every time.  The fact that computers can now play chess better than humans is based on only a few basic rules of logic.  Computers are composed of millions of microscopic switches that may be either on or off.  The **binary number system** with only 2 digits, **0 and 1** is well suited to represent the two switch states (on and off).  But where does the computer’s ability to make decisions come from?  If we let the switch represent True (on) and False (off) we can use a system of logic invented in the 1800's by an English mathematician named George Boole. He invented a mathematical way of looking at decision making logic that bears his name.  The most powerful computer is based on the concepts developed over 150 years ago by Boole.

### When is 1 and 1 not 2?

To understand Boole's logic let's look at a simple **series** circuit made up of 2 switches, a light, and a battery.  There are 4 possible states for this circuit configuration, shown in the next illustration:

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_1.png Figure Representation of the AND circuit

Note that there is only one configuration of switches that turns on the lamp.  The lamp is on only when SW1 **and** SW2 are on.  AND is called a Boolean operator and the rules for its use can be listed in a diagram called a **truth table**.  Some of the conventions used are: **on** can be represented by **true**, **1,** or **high**; and **off** by **false**, **0**, or **low**.  True and False are of course direct opposites.  0 and 1 are the only digits in the binary number system, and high and low refer to the electrical voltage in a computer circuit.  Therefore we can say that 1 **and** 1 is 1!

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_3.png Figure AND truth table

Although this seems simple enough, the AND concept is a very powerful tool.  A Venn diagram is another way mathematicians represent the AND operator.  For example, let’s think about searching through documents on the internet.  If you are looking for documents that contain both the words Jet Aircraft **and** Passenger Planes we can represent this by:

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_4.png Figure AND Venn diagram

The green circle represents all the documents that contain the term Jet Aircraft some of which are passenger planes.  The yellow circle represents all the documents that contain the term Passenger Planes some of which are jet aircraft.  The red section represents only those documents that contain both the terms.  In a search, these are the documents that would be listed.

### *Or* is there another possibility?

Another Boolean operator is **OR**. Consider the following circuit with the lamp and battery, but this time the 2 switches are in a parallel configuration:

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_2.png Figure Representation of the OR circuit

Tracing the circuit you can easily see that the lamp will be on if either **or** both of the switches are closed.  A truth table can be constructed and is illustrated next:

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_5.png Figure OR truth table

The OR operator applied to our internet search example above would return all the documents that contained either jet aircraft **or** passenger planes.  The Venn diagram would look like the following.

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_6.png Figure OR Venn diagram

### NOT!

The final basic Boolean operator is **NOT** (there are other operators but they are a combination of these 3).  Consider the following circuit:

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_7.png Figure Representation of the NOT circuit

The circuit above is not one you would want to construct in the manner shown (more on this later\*).  The switch when closed actually causes a short circuit as the electric current will flow through its very low resistance and not through the lamp.  The lamp will light only when the switch is off.

The truth table is much simpler for the NOT operator:

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_8.png Figure Not truth table

The NOT is also called an inverter in that the input and output are always exactly opposite, or complements of each other.  The Venn diagram would look like the following:

###### http://www.cdli.ca/courses/isys1205/phidgets/unit02_org01_ilo03/images/u02s01l03_9.png Figure NOT Venn diagram

\*In reality, the battery would be drained quickly.  NOT circuits are usually made with the tiny transistor switches that make up a computer. The circuit shown above simply serves to illustrate how the NOT operator works.

A computer is made of combinations of these fundamental logic circuits or logic **gates**.  Its ability to compute, compare, process characters, communicate over networks,  play music, and display stunning images and video is based on the relations worked out by George Boole.  As you will learn in a later lesson, his logic can be expressed in electric circuits and in programming languages.

Logic gates are the fundamental building blocks of digital electronics.  In circuit design they are given symbols and rules.  In the examples below, A and B are inputs to the gate and Q is the output.  In a digital device (eg. a computer) a high electrical voltage (eg. +5 volts) represents a 1 or true and a low voltage (0 volts) represents a 0 or false.

###### **See also the hand-out on logical gate operators**