## **Question: Who is George Boole and what does he have to do with the bool type?**

George Boole was an Anglo-Irish mathematician who created Boolean Logic, the mathematics behind modern computing, hardware and software. In computer programming languages, the Boolean type (this is bool in C#) can have only 2 values, true or false. You can use boolean variables to make decisions.

**Question: So, why do I want to know about the bool type in C#?**

You can use it to make decisions in your code!

**Question: OK, makes sense, how do I use a conditional in C#?**

BTW, sometimes ‘if’ gets substituted for ‘conditional’.

The most obvious form of the conditional is the if statement, and there are 3 variations of this. You can make decisions with a switch statement as well.

Suppose we have 2 integer variables, ageOfAdeline and ageOfJerome (exquisitely camel cased), here is variation #1, a single if…

if (ageOfAdeline > ageOfJerome)

Console.WriteLine(“Addie’s older than Jerri”);

We can also use a boolean variable to store a boolean value indicating this condition

bool isOlderAddie = ageOfAdeline > ageOfJerome;

… and then use this variable in the condition.

if (isOlderAddie)

Console.WriteLine(“Addie is older”);

We can use an ‘if’ with branches,

if (condition1) {

// Do this code

}

else {// Here, else would be the default if condition1 was not true

// Do some other code

}

You can have multiple branches…

if (condition1) {

// Do this code

}

else if (condition2) {

// Do code for condition 2

}

else if (condition3)

…..

else { // Here, the single else is the default and will happen if none of the

// preceding conditions are true

If a variable can only have 1 value, and typically that is indeed the case, you should use an ‘if’ with ‘else if’ branches, rather than a stream of ifs. Why? Although we have super fast processors, let’s not make the computer evaluate conditions it does not need to evaluate.

**Question: What’s a switch statement?**

A switch statement can be used to make decisions too, similar to an if with multiple if else branches. Here’s the anatomy of a switch statement, where someValue is compared to firstValue etc:

switch (someValue) {

case firstValue:

// Do code for if someValue == firstValue …

break;

case secondValue:

// Do code for if someValue == secondValue …

break;

default:

// Do code for the default case, someValue not equal to any of the values listed

break;

}

**Question: When should I use If with branches? When should I use switch?**

Entirely dependent on the circumstances, and also on personal choice. On the job, it may also depend on project conventions.

**Question: What if my condition has more than one part?**

If your condition has more than one part, then you may need to use a compound condition. This is a condition composed of more than one part both of which need to be evaluated, the parts being joined together with a logic operator, like AND (in C# is is denoted as &&), OR (in C# this is ||). There are truth tables for boolean operators, that specify the result of combining expression with AND (&&), OR (||) and also NOT (!).

Here is an example,

if ((ageOfAdeline > ageOfJerome) && (ageOfAdeline >= 21) {

// Code here for both conditions being true

}

You might have an either or situation,

if ((ageOfAdeline > ageOfJerome) || (ageOfAdeline >= 21) {

// Code here for either one, or both, of the conditions being true

}