## **Order Of Operations**

**The goal for this exercise** is to understand what order operators are executed in, so that you can both understand existing, and create your own, complex mathematical expressions

Microsoft has published a complete language specification for C#, which includes a table listing the precedence of each operator at:  
<http://msdn.microsoft.com/en-us/library/aa691323(v=vs.71).aspx>.

This should closely resemble what you've seen in your math classes (PEMDAS and all that), with some exceptions. Note also that while you're not expected to recognize all the operators (for example, the bitwise ones are probably new), you should recognize a lot of them from Java / JavaScript (the logical AND/OR operators). This topic is explained in more detail in your textbook (Section 3.5 == §3.5).

Given each of the following expressions, what order will the operators be evaluated in, and why? After each operator is evaluated, re-write the expression, with any appropriate values taking the place of the operator.

**What you need to do for this exercise:**  Fill in the following table, as demonstrated in the first cell. In terms of handing this in, you have two options: hand in this file by saving it in the directory that you .ZIP (that you send to the ‘Compressed Folders’). Or, write out your answer in the C# source code file, in comments, inside the class named Order\_Of\_Operations. Either one is fine, but DO NOT DO BOTH – choose just one.

|  |  |
| --- | --- |
| **Expression** | **Your Answer:** |
| 2 + 3 \* 2 | EXAMPLE: Multiplication first 🡪 2 + 6  Addition second 🡪 8 |
| 2 \* 6 / 3 |  |
| 20 – 6 / 2 |  |
| 43 > 20 && 20 > 10 |  |
| int x = 43;  int y = 20;  bool z = (x >= y && y < x); |  |
| int x = 2;  int y = 2 \*x + 1; |  |
| int x = 2;  int y = 2 \* (x + 1); |  |
| int x = 2;  bool z = 2 \* x >= 10; |  |

## 