# DATA MANIPULATION\*

## Kenneth Leroy Busbee

This work is produced by OpenStax-CNX and licensed under the Creative Commons Attribution License  $2.0^{\dagger}$ 

#### Abstract

An introduction to expressions with definitions, an example and an explanation of precedence.

## 1 Introduction

Single values by themselves are important; however we need a method of manipulating values (processing data). Scientists wanted an accurate machine for manipulating values. They wanted a machine to process numbers or calculate answers (that is compute the answer). Prior to 1950, dictionaries listed the definition of computers as "humans that do computations". Thus, all of the terminology for describing data manipulation is math oriented. Additionally, the two fundamental data type families (the integer family and floating-point family) consist entirely of number values.

## 2 Definitions

#### **Definition 1: expression**

A valid sequence of operand(s) and operator(s) that reduces (or evaluates) to a single value.

### **Definition 2: operator**

A language-specific syntactical token (usually a symbol) that causes an action to be taken on one or more operands.

## Definition 3: operand

A value that receives the operator's action.

#### **Definition 4: precedence**

Determines the order in which the operators are allowed to manipulate the operands.

#### **Definition 5: associativity**

Determines the order in which the operators of the same precedence are allowed to manipulate the operands.

#### **Definition 6: evaluation**

The process of applying the operators to the operands and resulting in a single value.

#### **Definition 7: parentheses**

Change the order of evaluation in an expression. You do what's in the parentheses first.

<sup>\*</sup>Version 1.6: Jan 15, 2010 6:06 am -0600

<sup>&</sup>lt;sup>†</sup>http://creativecommons.org/licenses/by/2.0/

## 3 An Expression Example with Evaluation

Let's look at an example: 2 + 3 \* 4 + 5 is our expression but what does it equal?

- 1. the symbols of + meaning addition and \* meaning multiplication are our operators
- 2. the values 2, 3, 4 and 5 are our operands
- 3. precedence says that multiplication is higher than addition
- 4. thus, we evaluate the 3 \* 4 to get 12
- 5. now we have: 2 + 12 + 5
- 6. the associativity rules say that addition goes left to right, thus we evaluate the 2 + 12 to get 14
- 7. now we have: 14 + 5
- 8. finally, we evaluate the 14 + 5 to get 19; which is the value of the expression

Parentheses would change the outcome. (2 + 3) \* (4 + 5) evaluates to 45.

Parentheses would change the outcome. (2 + 3) \* 4 + 5 evaluates to 25.

## 4 Precedence of Operators Chart

Each computer language has some rules that define precedence and associativity. They often follow rules we may have already learned. Multiplication and division come before addition and subtraction is a rule we learned in grade school. This rule still works. The precedence rules vary from one programming language to another. You should refer to the reference sheet that summarizes the rules for the language that you are using. It is often called a Precedence of Operators Chart. You should review this chart as needed when evaluating expressions.

A valid expression consists of operand(s) and operator(s) that are put together properly. Why the (s)? Some operators are:

- 1. Unary that is only have one operand
- 2. Binary that is have two operands, one on each side of the operator
- 3. Trinary which has two operator symbols that separate three operands

Most operators are binary, that is they require two operands. Within C++ there is only one trinary operator, the conditional. All of the unary operators are on the left side of the operand, except postfix increment and postfix decrement. Some precedence charts indicate of which operators are unary and trinary and thus all others are binary.