5. Expressions

In Cb, expressions consist of one or more operators in tandem with operands. Associativity rules determine precedence, but parentheses can override the default orderings. The two most pervasive expressions in Cb are assignment expressions and operation expressions. The table below outlines the associativity rules of the Cb’s built in functions.

|  |  |  |  |
| --- | --- | --- | --- |
| Tokens  (Descending Priority) | Operators | Class | Associativity |
| Identifiers, constants, parenthesized expression | Primary expression | Primary |  |
| () [] . | Function calls, subscripting, direct selection | Postfix | L-R |
| + - | Add/Minus | Binary | L-R |
| is isnt | Equality comparisons | Binary | L-R |
| < <= >= > | Relational | Binary | L-R |
| And | Logical And | Binary | L-R |
| Or | Logical Or | Binary | L-R |
| = | Assigment | Binary | R-L |

5.1 Primary expressions

5.1.1 Identifier

An identifier typifies a primary expression. Its declaration calls for the specification of a type of the identifier followed by the value of the identifier. It can refer to an lvalue or a function designator.

5.1.2 Constant

An integer, decimal, character, or floating constant is a primary expression of constant value. The capitalized letters A-G are constant expressions that each represent Notes of default duration having pre-defined values associated with the notes A-G, respectively. Naturally, Note constants are the most frequent example of constants in Cb.

5.1.3 Parenthesized expression

A parenthesized expression is a primary expression of the form ( expression ). It can be used to override precedence. For example, consider the two expressions below.

Expression 1: (note1 > note2) and (note3 < note2 or note3 < note1)

Expression 2: (note1 > note2 and note3 < note2) or (note3 < note1)

While the former will “and” the two subexpressions together, the latter will instead apply the “or” operator to the result.

5.2 Postfix

Postfix calls can be function calls, subscripts or direct selection. An example of each, respectively, is Note1.arpeggiate(2, 3), Chord1[3], and Chord1.length

5.2.1 Function Calls

A function call is characterized by a primary expression followed by parentheses enclosing an optional comma-seperated list of expressions. These expressions form the arguments to the function. Each and every function in Cb must be declared before it is called. The method signature must consist of [meth method\_name return\_type (argument-expression-list). The argument expression list may either be a single argument or a list of arguments. Additionally, the return argument must match the return type in the method signature.

A copy of each parameter is created in advance of the function call. As result, Cb uses argument-passing by value. Although a function may change the values of the parameters, the changes will not affect the values of the parameters. Recursive function calls are honored in Cb.

5.2.2 Subscripting

Only Chords and Stanzas can be operated on using the subscripting operation. For example, subscripting applied to a Chord can be used to select a particular note. The subscript operator allows both retrieval and mutation of elements.

5.2.3 Direct Selection

Pitch and duration in objects of type Note and Chord can be changed through directly accessing the objects. For example, A.pitch += 2 will result in C. The same paradigm applies to duration objects as well. Direct selection can be applied to Stanza as well to access the length.

5.3 Unary Operations

5.3.1 Increment/Decrement Operations

Plus-plus (++) and minus-minus (--) operations of the form (expression)++ can be used for a variety of purposes. When applied to a Note, the plus-plus or minus-minus operator will augment or diminish the Note, respectively. Analogously, the plus-plus or minus-minus operator applied to a Chord will augment or diminish each of the constituent Notes.

5.4 Binary Operations

5.4.1 Add and Subtract

Add and subtract binary operations can be applied to a multitude of objects. In general, any object added to another object of the same type will result in the concatenation of the two objects. For example, the plus operator applied to a Chord or Stanza, the result is a concatenated or reduced sequence When applied to a Note, the Note is augmented or diminished by the argument of the expression. Chords can be added to Stanzas through the add and subtract methods but number literals cannot be added to chords, notes or stanzas.

The syntax is as follows:

Add-expression: add-expression + add-expression

Subtract-expression: subtract-expression – subtract-expression

5.4.2 Multiply, Divide and Modulus

Multiply can applied to Note, Chord, Scale and Stanza objects to create copies of the instance as well as to numbers to apply regular multiplication rules. Division and modulus can only be applied to real numbers.

The syntax for each of these expressions is analogous:

Multiply-expression: multiply-expression \* multiply-expression

Divide-expression: divide-expression / divide-expression

Modulus-expression: modoulus-expression % modulus-expression

5.4.3 Augmentation Operator

The augmentation operator (^) can be applied to a note to augment the note by a number of octaves or to a chord to transpose every constituent note by a number of octaves. Note ^ (Number) or Chord ^ (Number) exemplifies the syntax of the carrot operator. The range of allowable octaves for any note to assume the value of is -5 to 5.

5.4.5 Relational Comparisons

Yields a Number result (1 if true, 0 if false) that uses the following syntax:

Relational-expression:

relational-expression < relational-expression

relational-expression > relational-expression

relational-expression >= relational-expression

relational-expression <= relational-expression

5.4.6 Equality Comparisons

Determines if two values are equal. Cb uses 1 to denote true and 0 to denote false.

The token “is” denotes equality while “isnt” denotes inequality.

The following rules govern equality relations:

Two Number objects are equal if they have the same value.

Two Note objects are equal if they have the same duration and pitch.

Two Chord objects are equal if they consist of the same notes for the same duration

Two Stanza objects are equal if they have the same chords and notes in the same order.

Equality Comparisons take the following form:

Equality-expression is equality-expression

Equality-expression is not equality-expression

5.4.7 Logical Operators

“And” and “or” perform a logical and, or operation on two expressions, respectively. If the expression evaluates to false, then a zero is returned. Otherwise, 1 is returned. Lazy evaluations or “short-circuiting” is supported.

Logical-expression:

logical-and-expression and logical-and-expression

logical-or-expression or logical-or-expression

5.4.8 Assignment

Assignment is a right associative operation – the expression on the right is evaluated and then used to set the lvalue. The rvalue must have the same type as the lvalue since no casting is implicitly done.

5.4.9 Commas

Commas are used to separate list elements like parameters in a function or Notes in a Chord. Consider, for example, Chord chord = (noteA, noteB). Moreover, a pair of expressions separated by a comma is evaluated left-to-right and that the type and value of the result are identical to the type and value of the right operand.

5.4.10 Expressions of the form [Operation]-Equals

The tokens “+=”, “-=”, “/=”, “\*=” can be used to modify the state of a variable by a given amount. For example, A += 2 will return a Note of value C with a default duration. Each of the operators uses the pre-defined operations of addition, subtraction, division and multiplication to compute the result.