Ternary operation

From Wikipedia, the free encyclopedia

In mathematics, a **ternary operation** is an n-ary operation with n = 3. A ternary operation on a set A takes any given three elements of A and combines them to form a single element of A. An example of a ternary operation is the product in a heap.

In computer science a **ternary operator** (sometimes incorrectly called a *tertiary operator*) is an operator that takes three arguments. The arguments and result can be of different types.

Many programming languages that use C-like syntax feature a ternary operator, `?:`, which defines a conditional expression. Since this operator is often the only existing ternary operator in the language, it is sometimes simply referred to as "*the* ternary operator", though it is more accurately referred to as the **conditional operator**.

Most of the languages emphasizing functional programming don't need such an operator as their regular conditional expression(s) is an expression in the first place e.g. the Scheme expression `(if (> a b) a b)` is equivalent in semantics to the C expression `(a > b) ? a : b`. This is also the case in many imperative languages, starting with ALGOL where it is possible to write `result := if a > b then a else b`, or Smalltalk (`result := (a > b) ifTrue: [ a ] ifFalse: [ b ]`) or Ruby (`result = if a > b: a else b end`)

Contents

- 1 Examples
  - 1.1 C, C++, C#, Objective-C, Java, JavaScript, ActionScript
  - 1.2 Common Lisp
  - 1.3 Perl, PHP
  - 1.4 Perl 6
  - 1.5 Python
  - 1.6 Ruby
  - 1.7 Visual Basic
- 2 See also
- 3 References
- 4 External links

Examples

C, C++, C#, Objective-C, Java, JavaScript, ActionScript

A traditional if-else construct in C, C++, C#, Objective-C, Java, JavaScript and ActionScript is

written:[1]

```
if (a > b) {
    result = x;
}
else {
    result = y;
}
```

This can be rewritten as the following ternary statement:

```
result = a > b ? x : y;
```

Common Lisp

Assignment using a ternary expression in Common Lisp:

```
(setf result (if (a> b) x y))
```

Imperative form:

```
(if (a> b)
  (setf result x)
  (setf result y))
```

Perl, PHP

A traditional if-else construct in Perl or PHP is written:

```
if ($a > $b) {
    $result = $x;
} else {
    $result = $y;
}
```

Rewritten to use the ternary operator:

```
$result = ($a > $b) ? $x : $y;
```

As of version 5.3, PHP supports a ternary shortcut syntax `expr1 ?: expr2` which is equivalent to `expr1 ? expr1 : expr2`, i.e. the condition is returned unless it evaluates to false when converted to a boolean value.

```
  $result = $condition ?: $ifFalse;
```

Nested ternary operators are allowed in PHP (Perl) but are not always best practice and create code that is more difficult to read.[2] A traditional nested if-else statement in PHP is written:

```
if ($a > $b) {
        if ($a > $c) {
                $result = $a;
        } else {
                $result = $c;
        }
} else {
        if ($b > $c) {
                $result = $b;
        } else {
                $result = $c;
        }
}
```

Rewritten to use the nested ternary operator:

```
$result = ($a > $b) ? ($a > $c ? $a : $c) : ($b > $c ? $b : $c);
```

Perl 6

Uses double symbols and "!" instead of ":" [3]:

```
$result = ($a > $b) ?? $x !! $y;
```

Python

Though it had been delayed for several years by disagreements over syntax, a ternary operator for Python was approved as Python Enhancement Proposal 308 (http://www.python.org/dev/peps/pep-0308/) and was added to the 2.5 release in September 2006. Python's ternary operator differs from the common `?:` operator in the order of its operands. The general form is:

```
x if a > b else y
```

This form invites considering `x` as the normal value and `y` as an exceptional case. One can use the syntax

```
(lambda:y, lambda:x)[a > b]()
```

as a workaround for code that also needs to run under Python versions before 2.5. Note that operands are lazily evaluated, it is possible to remove the lambdas and function calls but the operands will be eagerly evaluated which isn't consistent with other languages' ternary operators, e.g. by using an explicitly constructed dictionary:

```
{True: x, False: y}[a > b]
```

A less reliable but simpler to read alternative was to abuse the `and` and `or` operators and write

```
(a > b) and x or y
```

but this code would break if `x` could be a "falsy" value (`None`, `False`, `0`, an empty sequence or

collection, …) as the expression would return `y` (whether it was truthy or falsy) instead of the (falsy) `x`. A possible workaround is to make `x` and `y` arrays, so they are never falsy, and then grab the first element of the resulting array as in the following

```
((a > b) and [x] or [y])[0]
```

Ruby

A traditional if-else construct in Ruby is written:[4]

```
if a > b
  result = x
else
  result = y
end
```

This could also be written as:

```
result = if a > b
  x
else
  y
end
```

These can be rewritten as the following ternary statement:

```
result = a > b ? x : y
```

Visual Basic

Visual Basic Version 9 has added a ternary operator, `If()`, in addition to the existing `IIf()` function that existed previously. As a true operator, it does not have the side effects and potential inefficiencies of the `IIf()` function.

The syntaxes of the tokens are similar: `If(condition[, op1], op2)` vs `IIf(condition, op1, op2)`. As mentioned above, the function call has significant disadvantages, because the subexpressions must all be evaluated, according to Visual Basic's evaluation strategy for function calls and the result will always be of type variant (VB) or object (VB.NET). The `If()`operator however does not suffer from these problems as it supports conditional evaluation and determines the type of the expression based on the types of its operands.

See also

- ?:, the ternary conditional expression
- IIf, inline if function

References

1. **^** Learning GNU C: The Conditional Operator (http://www.faqs.org/docs/learnc/x598.html)
2. **^** "Ternary Operators" (http://www.kwista.com/programming/ternary-operators/) . Kwista, LLC.. http://www.kwista.com/programming/ternary-operators/. Retrieved 2010-04-19.
3. **^** "Perl6 Operators" (http://perlcabal.org/syn/S03.html#Conditional_operator_precedence/) . Larry Wall. http://perlcabal.org/syn/S03.html#Conditional_operator_precedence/. Retrieved 2010-05-18.
4. **^** Programming Ruby: Conditional Execution (http://ruby-doc.org/docs/ProgrammingRuby/html/tut_expressions.html#S4)

External links

- Description of Ternary Operator in C# (http://msdn2.microsoft.com/en-us/library/ty67wk28(VS.80).aspx)
- Description of If operator in Visual Basic (http://msdn.microsoft.com/en-us/library/bb513985.aspx)
- Description of Ternary Conditional Expression in Python (PEP 308) (http://www.python.org/peps/pep-0308.html)
- Alternative workaround for ternary conditional expression (http://www.diveintopython.org/power_of_introspection/and_or.html)
- Description in the Java Language Specification (http://java.sun.com/docs/books/jls/third_edition/html/expressions.html#15.25)
- Description in the PHP Language Documentation (http://php.net/manual/en/language.operators.comparison.php#language.operators.comparison.ternary)
- Visual Basic 9.0 Language Specification (http://www.microsoft.com/downloads/details.aspx?familyid=39DE1DD0-F775-40BF-A191-09F5A95EF500&displaylang=en)

Retrieved from "http://en.wikipedia.org/wiki/Ternary_operation"
Categories: Ternary operations | Conditional constructs

- This page was last modified on 17 March 2011 at 10:09.
- Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. See Terms of Use for details.
  Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.