Search Wikipedia

Q



Main page Contents Featured content Current events Random article Donate to Wikipedia

Wikipedia store

Interaction Help About Wikipedia Community portal Recent changes

Contact page

Tools What links here Related changes Upload file Special pages Permanent link Page information Wikidata item

Print/export Download as PDF Printable version

Cite this page

Languages

中文

**★**A 9 more

Edit links

العربية Deutsch Español Français 한국어 日本語 Русский Svenska

0

Increment and decrement operators

From Wikipedia, the free encyclopedia

Talk

"++" redirects here. For the Loona EP, see ++ (EP).



This article **needs additional citations for verification**. Please help improve this article by adding citations to reliable sources. Unsourced material may be challenged and removed. Find sources: "Increment and decrement operators" – news · newspapers · books · scholar · JSTOR (September 2014) (Learn how and when to remove this template message)

Read

Edit View history

**Increment** and **decrement operators** are unary operators that *add* or *subtract* one, to or from their operand, respectively. They are commonly implemented in imperative programming languages. C-like languages feature two versions (pre- and post-) of each operator with slightly different semantics.

In languages syntactically derived from B (including C and its various derivatives), the increment operator is written as ++ and the decrement operator is written as -- . Several other languages use inc(x) and dec(x) functions.

The increment operator increases, and the decrement operator decreases, the value of its operand by 1. The operand must have an

arithmetic or pointer data type, and must refer to a modifiable data object. Pointers values are increased (or decreased) by an amount that makes them point to the next (or previous) element adjacent in memory.

In languages that support both versions of the operators:

- The *pre*-increment and *pre*-decrement operators increment (or decrement) their operand by 1, and the value of the expression is the resulting incremented (or decremented) value.
- The post-increment and post-decrement operators increase (or decrease) the value of their operand by 1, but the value of the expression is the operand's original value *prior* to the increment (or decrement) operation.

In languages where increment/decrement is not an expression (e.g. Go), only one version is needed (in the case of Go, post operators only).

Since the increment/decrement operator modifies its operand, use of such an operand more than once within the same expression can produce undefined results. For example, in expressions such as x - ++x, it is not clear in what sequence the subtraction and increment operations should be performed. Such expressions generally invoke undefined behavior, and should be avoided.

```
Contents [hide]
1 Examples
2 Supporting languages
3 History
4 See also
5 References
```

## Examples [edit] The following C code fragment illustrates the difference between the *pre* and *post* increment and decrement operators:

```
int x;
int y;
// Increment operators
// Pre-increment - x is incremented by 1, then y is assigned the value of x
y = ++x; // x is now 2, y is also 2
// Post-increment - y is assigned the value of x, then x is incremented by 1
x = 1;
y = x++; // x is now 2, y is 1
// Decrement operators
// Pre-decrement - x is decremented by 1, then y is assigned the value of x
y = --x; // x is now 0, y is also 0
// Post-decrement - y is assigned the value of x, then x is decremented by 1
x = 1;
y = x--; // x is now 0, y is 1
```

```
# Pre-increment: y = ++x
x = x + 1 # x is now 2 (can be written as "x += 1" in Python)
y = x # y is also 2
# Post-increment: y = x++
\mathbf{x} = 1
y = x  # y is 1
x = x + 1 \# x \text{ is now } 2
```

The post-increment operator is commonly used with array subscripts. For example:

In languages lacking these operators, equivalent results require an extra line of code:

```
// Sum the elements of an array
float sum elements(float arr[], int n)
   float sum = 0.0;
          i = 0;
   int
   while (i < n)
       sum += arr[i++];  // Post-increment of i, which steps
                          // through n elements of the array
   return sum;
}
```

// Copy one array to another

The post-increment operator is also commonly used with pointers:

```
void copy array(float *src, float *dst, int n)
     while (n-->0) // Loop that counts down from n to zero
          *dst++ = *src++; // Copies element *(src) to *(dst),
                             // then increments both pointers
 }
Note that these examples also work in other C-like languages, such as C++, Java, and C#.
```

• Increment operator can be demonstrated by an example:

```
#include <stdio.h>
int main()
```

```
int c=2, d=2;
  printf("%d\n", c++); // this statement displays 2 then, only c incremented by 1 to 3.
  printf("%d", ++c); // this statement increments 1 to c then, only c is displayed.
  return 0;
• Output:
```

2 4

```
Supporting languages [edit]
The following list, though not complete or all-inclusive, lists some of the major programming languages that support the ++ / --
```

## increment/decrement operators. • D<sup>[6]</sup> ABAP

 PARI/GP<sup>[7]</sup> AWK<sup>[1]</sup> Go Perl

```
    Bash<sup>[2]</sup>

                                                                                      PHP
                                            Java
  • C<sup>[3]</sup>

    PowerShell<sup>[8]</sup>

    JavaScript

 • C++<sup>[4]</sup>

    Objective-C

                                                                                      Vala
 • C#<sup>[5]</sup>

    GNU Octave

    Vex, a scripting language in the software Houdini

 CFML

    Wolfram Language<sup>[9]</sup>

(Apple's Swift once supported these operators, [10] but support was removed as of version 3.)
Pascal, Delphi, Modula-2, and Oberon provide the same functions, but they are called inc(x) and dec(x).
```

History [edit]

The concept was introduced in the B programming language circa 1969 by Ken Thompson.<sup>[11]</sup> Thompson went a step further by inventing the ++ and -- operators, which increment or decrement; their prefix or postfix position determines whether the alteration occurs before or after noting the value of the operand. They were not in the earliest versions

address modes provided by the DEC PDP-11 on which C and Unix first became popular. This is historically impossible, since there was no PDP-11 when B was developed. The PDP-7, however, did have a few 'auto-increment' memory cells, with the property that an indirect memory reference through them incremented the cell. This feature probably suggested such operators to Thompson; the generalization to make them both prefix and postfix was his own. Indeed, the auto-increment cells were not used directly in implementation of the operators, and a stronger motivation for the innovation was probably his observation that the translation of ++x was smaller than that of x=x+1. See also [edit] Augmented assignment – for += and -= operators

of B, but appeared along the way. People often guess that they were created to use the auto-increment and auto-decrement

- PDP-7 • PDP-11
- References [edit]

Successor function

## 1. ^ "GNU Awk's User Guide" &. Free Software Foundation.

```
2. ^ "8.3. The Double-Parentheses Construct" 2. The Linux
  Documentation Project.
3. A Ritchie, Brian W. Kernighan; Dennis M.; Ritchie, Dennis (1988). The
```

- C programming language (2. ed., [Nachdr.] ed.). Englewood Cliffs, N.J.: Prentice Hall. p. 18 ₽. ISBN 0-13-110362-8. 4. ^ "Increment/decrement operators" ☑. cppreference.com.
- 5. ^ "++ Operator (C# Reference)" ☑. Microsoft Developer Network.
- 8. ^ "About Assignment Operators" &. 9. ^ "Increment Wolfram Language Symbol" ☑. Wolfram Language

Documentation Center.

7. ^ "GP Operators and their Priorities" &.

- 10. ^ "Basic Operators" ☑. developer.apple.com. 11. A Ritchie, Dennis M. (March 1993). "The Development of the C
- Language" ☑. ACM SIGPLAN Notices. 28 (3): 5.

Powered By MediaWiki

WIKIMEDIA

Categories: Operators (programming) | Unary operations

This page was last edited on 19 May 2020, at 15:13 (UTC).

Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.

```
Privacy policy About Wikipedia Disclaimers Contact Wikipedia Developers Statistics Cookie statement Mobile view
```