Операторы в Java

Операторы в языке Java - это специальные символы, которые сообщают транслятору о том, что вы хотите выполнить операцию с некоторыми операндами. Все операторы в Java полностью аналогичны операторам в языке С++. В языке С++ нет только операторов сдвига с заполнением нулями.

Далее будет идти речь об операциях с простыми типами. Операторы могут быть переопределены для любых типов. Например, существует оператор «+» для строк.

Операторы, требующие одного операнда, называются *унарными*. Одни операторы ставятся перед операндами и называются *префиксными*, другие - после, их называют *постфиксными* операторами.

Большинство же операторов ставят между двумя операндами, такие операторы *бинарными* операторами. Существует также тернарный опе­ратор, работающий с тремя операндами.

|  |  |  |  |
| --- | --- | --- | --- |
| Операторы | | | |
| Бинарные | Унарные | | Тернарный |
| Префиксными | Постфиксными |

Операторы в Java можно разбить на такие классы:

|  |  |  |  |
| --- | --- | --- | --- |
| Операторы | | | |
| Арифметические | Битовые | Операторы сравнения | Логические |

**Арифметические операторы**

Арифметические операторы используются для вычислений так же как в алгебре. Допустимые операнды должны иметь числовые типы. Например, исполь­зовать эти операторы для работы с логическими типами нельзя.

|  |  |  |  |
| --- | --- | --- | --- |
| Оператор | Результат | Оператор | Результат |
| + | Сложение | + = | сложение с присваиванием |
| - | вычитание (также унарный минус) | -= | вычитание с присваиванием |
| \* | Умножение | \* = | умножение с присваиванием |
| / | Деление | /= | деление с присваиванием |
| % | деление по модулю | %= | деление по модулю с присваиванием |
| ++ | Инкремент | -- | декремент |

**Целочисленные битовые операторы**

Для целых числовых типов данных - long, int, short, char и byte, определен дополнительный набор операторов, с помощью которых можно проверять и модифицировать состояние отдельных битов соответствую­щих значений. Операторы битовой арифметики работают с каждым битом как с самостоятельной величиной.

|  |  |  |  |
| --- | --- | --- | --- |
| Оператор | Результат | Оператор | Результат |
| ~ | побитовое унарное отрицание (NOT) |  |  |
| & | побитовое И (AND) | &= | побитовое И (AND) с присваиванием |
| | | побитовое ИЛИ (OR) | |= | побитовое ИЛИ (OR) с присваиванием |
| ^ | побитовое исключающее ИЛИ (XOR) | ^= | побитовое исключающее ИЛИ (XOR) с присваиванием |
| >> | сдвиг вправо | >> = | сдвиг вправо с присваиванием |
| >>> | сдвиг вправо с заполнением нулями | >>>= | сдвиг вправо с заполнением нулями с присваиванием |
| << | сдвиг влево | <<= | сдвиг влево с присваиванием |

Результат работы побитовых операторов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| А | В | OR | AND | XOR | NOT A |
| 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 | 0 |

**Операторы отношения**

|  |  |
| --- | --- |
| Оператор | Результат |
| == | равно |
| != | не равно |
| > | больше |
| < | меньше |
| >= | больше или равно |
| <= | меньше или равно |

Для того, чтобы можно было сравнивать два значения, в Java имеется набор операторов, описывающих отношение и равенство. Список таких операторов приведен в таблице.

Значения любых типов, включая целые и вещественные числа, сим­волы, логические значения и ссылки, можно сравнивать, используя опе­ратор проверки на равенство == и неравенство !=.

В языке Java, так же, как в С и C++ проверка на равенство обознача­ется последовательностью (==). Один знак (=) - это оператор присваи­вания.

**Булевы логические операторы**

Булевы логические операторы, сводка которых приведена в таблице ниже, оперируют только с операндами типа boolean. Все бинарные ло­гические операторы воспринимают в качестве операндов два значения типа boolean и возвращают результат того же типа.

|  |  |  |  |
| --- | --- | --- | --- |
| Оператор | Результат | Оператор | Результат |
| & | логическое И (AND) | &= | И (AND) с присваиванием |
| | | логическое ИЛИ (OR) | |= | ИЛИ (OR) с присваиванием |
| ^ | логическое исключающее ИЛИ (XOR) | ^= | исключающее ИЛИ (XOR) с присваиванием |
| || | оператор OR быстрой оценки выражений (short circuit OR) | == | равно |
| && | оператор AND быстрой оценки выражений (short circuit AND) | != | не равно |
| ! | логическое унарное отрицание (NOT) | ?: | тернарный оператор if-then-else |

Результаты воздействия логических операторов на различные комби­нации значений операндов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| А | В | OR | AND | XOR | NOT A |
| false | false | false | false | false | true |
| true | false | true | false | true | false |
| false | true | true | false | true | true |
| true | true | true | true | false | false |

**Операторы быстрой оценки логических выражений (short circuit logical operators)**

Существуют два интересных дополнения к набору логических опера­торов. Это - альтернативные версии операторов AND и OR, служащие для быстрой оценки логических выражений.

Если первый операнд оператора OR имеет значение true, то независимо от значения второго операнда результатом операции будет величина true. Аналогично в случае оператора AND, если первый операнд — false, то значение вто­рого операнда на результат не влияет - он всегда будет равен false.

Если используются операторы && и || вместо обычных форм & и |, то Java не производит оценку правого операнда логического выражения, если ответ ясен из значения левого операнда.

Общепринятой практикой является использование операторов && и || практически во всех случаях оценки булевых логических выражений. Версии этих операторов & и | применяются только в битовой арифметике.

**Тернарный оператор if-then-else**

Общая форма оператора if-then-use такова:

**выражение1? выражение2: выражениеЗ**

В качестве первого операнда — «выражение1» — может быть исполь­зовано любое выражение, результатом которого является значение типа boolean. Если результат равен true, то выполняется оператор, заданный вторым операндом, то есть, «выражение2». Если же первый операнд paвен false, то выполняется третий операнд — «выражениеЗ». Второй и третий операнды, то есть «выражение2» и «выражениеЗ», должны воз­вращать значения одного типа и не должны иметь тип void.