В языке программирования Java существует несколько типов операторов, которые выполняют различные операции с данными. Вот основные операторы:

**Арифметические операторы**

Арифметические операторы используются для выполнения математических операций.

* + : сложение
* - : вычитание
* \* : умножение
* / : деление
* % : остаток от деления (модуль)

Пример:

int a = 10;

int b = 3;

int sum = a + b; // 13

int diff = a - b; // 7

int product = a \* b; // 30

int quotient = a / b; // 3

int remainder = a % b; // 1

**Операторы сравнения**

Операторы сравнения используются для сравнения двух значений и возвращают логическое значение true или false.

* == : равно
* != : не равно
* > : больше
* < : меньше
* >= : больше или равно
* <= : меньше или равно

Пример:

int a = 10;

int b = 20;

boolean result1 = a == b; // false

boolean result2 = a != b; // true

boolean result3 = a > b; // false

boolean result4 = a < b; // true

boolean result5 = a >= b; // false

boolean result6 = a <= b; // true

**Логические операторы**

Логические операторы используются для выполнения логических операций, обычно с булевыми значениями.

* && : логическое "И" (AND)
* || : логическое "ИЛИ" (OR)
* ! : логическое "НЕ" (NOT)

Пример:

boolean a = true;

boolean b = false;

boolean result1 = a && b; // false (оба операнда должны быть true)

boolean result2 = a || b; // true (хотя бы один операнд должен быть true)

boolean result3 = !a; // false (отрицание)

**Операторы присваивания**

Операторы присваивания используются для присваивания значений переменным.

* = : присваивание
* += : присваивание с добавлением (эквивалентно a = a + b)
* -= : присваивание с вычитанием
* \*= : присваивание с умножением
* /= : присваивание с делением
* %= : присваивание с взятием остатка от деления

Пример:

int a = 10;

a += 5; // a = a + 5, теперь a = 15

a -= 3; // a = a - 3, теперь a = 12

a \*= 2; // a = a \* 2, теперь a = 24

a /= 4; // a = a / 4, теперь a = 6

a %= 3; // a = a % 3, теперь a = 0

**Операторы инкремента и декремента**

Эти операторы изменяют значение переменной на единицу.

* ++ : инкремент (увеличивает на 1)
* -- : декремент (уменьшает на 1)

Пример:

int a = 5;

a++; // a = a + 1, теперь a = 6

a--; // a = a - 1, теперь a = 5

Также существует форма **префиксная** (++a, --a) и **постфиксная** (a++, a--) инкремента/декремента:

* Префиксная форма сначала изменяет значение, затем используется в выражении.
* Постфиксная форма сначала использует значение в выражении, а затем изменяет.

**Тернарный оператор**

Тернарный оператор — это сокращённая форма для условных выражений. Он имеет следующий синтаксис:

* condition ? value\_if\_true : value\_if\_false

Пример:

int a = 5;

int b = (a > 10) ? 100 : 200; // b = 200 (потому что a > 10 неверно)

**Операторы побитовых операций**

Побитовые операторы работают с битами целых чисел.

* & : побитовое "И"
* | : побитовое "ИЛИ"
* ^ : побитовое "ИСКЛЮЧАЮЩЕЕ ИЛИ" (XOR)
* ~ : побитовое "НЕ" (инвертирование всех битов)
* << : побитовый сдвиг влево
* >> : побитовый сдвиг вправо
* >>> : логический побитовый сдвиг вправо (заполняет нулями)

Пример:

int a = 5; // 0101 в бинарном виде

int b = 3; // 0011 в бинарном виде

int c = a & b; // 0001 (побитовое "И")

int d = a | b; // 0111 (побитовое "ИЛИ")

int e = a ^ b; // 0110 (побитовое "ИСКЛЮЧАЮЩЕЕ ИЛИ")

int f = ~a; // 1010 (побитовое "НЕ", инвертирует все биты)

int g = a << 1; // 1010 (сдвиг влево на 1)

int h = a >> 1; // 0010 (сдвиг вправо на 1)

**Операторы типа instanceof**

Оператор instanceof используется для проверки, является ли объект экземпляром определённого класса или интерфейса.

Пример:

String str = "Hello";

boolean result = str instanceof String; // true

**Операторы вызова метода и доступа к полям**

Используются для доступа к полям и методам объектов.

* . : доступ к полям и методам объекта

Пример:

String str = "Hello";

int length = str.length(); // вызов метода length()

char ch = str.charAt(0); // доступ к символу строки по индексу

**Старшинство операций**

1. () {} .

2. ++ -- ~ !

3. \* / %

4. + -

5. » >>>> «

6. > >= < <=

7. == !=

8. &

9. ^

10. |

11. &&

12. ||

13. ?:

14. = op=