6.演算子

Javaには、様々な種類の演算子があります。主な演算子には、以下のようなものがあります。

1. 算術演算子：+、-、\*、/、%。これらの演算子は、数値の加算、減算、乗算、除算、剰余（余り）を計算します。
2. 比較演算子：==、!=、>、<、>=、<=。これらの演算子は、2つのオペランドを比較し、真偽値を返します。
3. 論理演算子：&&、||、!。これらの演算子は、真偽値を操作し、真偽値を返します。
4. 代入演算子：=、+=、-=、\*=、/=、%=、&=、|=、^=、<<=、>>=、>>>=。これらの演算子は、左辺の変数に右辺の値を代入し、結果を左辺の変数に格納します。
5. インクリメント/デクリメント演算子：++、--。これらの演算子は、変数の値を1つ増やす（インクリメント）または1つ減らす（デクリメント）操作を行います。
6. ビット演算子：&、|、^、~、<<、>>、>>>。これらの演算子は、ビット単位での論理演算、シフト、反転を行います。
7. 条件演算子（三項演算子）：? :。これは、if-else文を簡略化したもので、条件式が真の場合は式1を、偽の場合は式2を返します。

これらの演算子は、Javaプログラムで頻繁に使用されます。適切に使用することで、より効率的なコードを記述することができます。

6.1算術演算子

Javaの算術演算子は、以下の5つです。

1. 加算演算子（+）：2つの値を加算します。 例：int a = 5 + 3; // aに6が代入されます。
2. 減算演算子（-）：2つの値を減算します。 例：int a = 5 - 3; // aに2が代入されます。
3. 乗算演算子（\*）：2つの値を乗算します。 例：int a = 5 \* 3; // aに15が代入されます。
4. 除算演算子（/）：左辺の値を右辺の値で除算します。 例：int a = 10 / 2; // aに5が代入されます。
5. 剰余演算子（%）：左辺の値を右辺の値で除算し、余りを返します。 例：int a = 10 % 3; // aに1が代入されます。

以下に、Javaでの数値の加算、減算、乗算、除算、剰余（余り）を計算するサンプルコードを示します。

ublic class ArithmeticOperations {

public static void main(String[] args) {

int a = 10;

int b = 3;

// 加算

int sum = a + b;

System.out.println("加算結果：" + sum); // 出力結果：13

// 減算

int difference = a - b;

System.out.println("減算結果：" + difference); // 出力結果：7

// 乗算

int product = a \* b;

System.out.println("乗算結果：" + product); // 出力結果：30

// 除算

double quotient = (double) a / b;

System.out.println("除算結果：" + quotient); // 出力結果：3.3333333333333335

// 剰余（余り）

int remainder = a % b;

System.out.println("剰余（余り）の結果：" + remainder); // 出力結果：1

}

}

上記のコードでは、**a**と**b**という2つの整数を使用して、加算、減算、乗算、除算、剰余（余り）の演算を行っています。それぞれの結果を変数に代入して、**System.out.println()**メソッドを使用して、結果をコンソールに出力しています。

なお、除算の結果は通常、整数型の場合は整数の商が返されますが、上記のコードでは**quotient**変数をdouble型にキャストすることで、小数点以下の値も含めた正確な商を計算しています。

6.2比較演算子？

比較演算子とは、Javaなどのプログラミング言語で、2つの値を比較するために使用される演算子のことを指します。比較演算子は、2つの値を比較して、真偽値（true/false）を返します。

以下は、Javaで使用される比較演算子の一覧です。

| **演算子** | **説明** |
| --- | --- |
| == | 左辺と右辺の値が等しい場合にtrueを返す |
| != | 左辺と右辺の値が異なる場合にtrueを返す |
| > | 左辺の値が右辺の値より大きい場合にtrueを返す |
| >= | 左辺の値が右辺の値以上の場合にtrueを返す |
| < | 左辺の値が右辺の値より小さい場合にtrueを返す |
| <= | 左辺の値が右辺の値以下の場合にtrueを返す |

以下に、比較演算子の使用例を示します。

int a = 10;

int b = 5;

if (a == b) {

System.out.println("aとbは等しい");

} else {

System.out.println("aとbは等しくない");

}

if (a != b) {

System.out.println("aとbは等しくない");

} else {

System.out.println("aとbは等しい");

}

if (a > b) {

System.out.println("aはbよりも大きい");

} else {

System.out.println("aはbよりも小さいか等しい");

}

if (a >= b) {

System.out.println("aはb以上");

} else {

System.out.println("aはb未満");

}

if (a < b) {

System.out.println("aはbよりも小さい");

} else {

System.out.println("aはbよりも大きいか等しい");

}

if (a <= b) {

System.out.println("aはb以下");

} else {

System.out.println("aはbよりも大きい");

}

上記のコードでは、変数**a**と**b**にそれぞれ10と5を代入し、比較演算子を使用してそれらの値を比較しています。比較結果に応じて、条件分岐を行っています。実行すると、以下のような出力が得られます。

aとbは等しくない

aとbは等しくない

aはbよりも大きい

aはb以上

aはbよりも大きいか等しい

aはb未満

6.3論理演算子

Javaの論理演算子には、以下の3種類があります。

1. AND演算子(&&, and): 2つの条件式が両方ともtrueの場合にtrueを返します。両側の式がfalseの場合はfalseを返します。例えば、**if (a > 0 && b > 0)**のように使います。
2. OR演算子(||, or): 2つの条件式のいずれかがtrueの場合にtrueを返します。両側の式がfalseの場合はfalseを返します。例えば、**if (a > 0 || b > 0)**のように使います。
3. NOT演算子(!, not): 条件式の結果を反転させます。条件式がtrueの場合はfalseを、falseの場合はtrueを返します。例えば、**if (!(a > b))**のように使います。

これらの論理演算子は、条件分岐やループ処理などで複数の条件を判定する場合に使用します。例えば、AND演算子を使って2つの条件が共に成り立つ場合に処理を実行する、OR演算子を使って2つの条件のいずれかが成り立つ場合に処理を実行する、NOT演算子を使って条件を反転させるなどがあります。

以下にJavaの論理演算子のサンプルコードを示します。

public class LogicalOperatorsExample {

public static void main(String[] args) {

int a = 10;

int b = 20;

int c = 30;

// AND演算子

if (a > 0 && b > 0) {

System.out.println("aとbは共に正の数です");

}

// OR演算子

if (a > 0 || b > 0) {

System.out.println("aまたはbは正の数です");

}

// NOT演算子

if (!(a > b)) {

System.out.println("aはbより小さいです");

}

// 複数の条件式を組み合わせた例

if (a > 0 && b > 0 || c > 0) {

System.out.println("aとbのいずれかは正の数、またはcは正の数です");

}

}

上記の例では、**&&**演算子を使ったAND演算、**||**演算子を使ったOR演算、**!**演算子を使ったNOT演算を行っています。また、複数の条件式を組み合わせた例も示しています。それぞれの演算子を使って、条件分岐やループ処理などで複数の条件を判定することができます。}

6.4 代入演算子

Javaには様々な種類の代入演算子があります。代入演算子は、変数に値を代入する際に使用されます。以下にJavaの代入演算子の一覧を示します。

1. 代入演算子(=): 右辺の値を左辺の変数に代入します。例えば、**int a = 10;**のように使います。
2. 加算代入演算子(+=): 左辺の変数に右辺の値を加え、結果を左辺の変数に代入します。例えば、**a += 5;**は、**a = a + 5;**と同じです。
3. 減算代入演算子(-=): 左辺の変数から右辺の値を引き、結果を左辺の変数に代入します。例えば、**a -= 5;**は、**a = a - 5;**と同じです。
4. 乗算代入演算子(\*=): 左辺の変数に右辺の値を掛け、結果を左辺の変数に代入します。例えば、**a \*= 5;**は、**a = a \* 5;**と同じです。
5. 除算代入演算子(/=): 左辺の変数を右辺の値で割り、結果を左辺の変数に代入します。例えば、**a /= 5;**は、**a = a / 5;**と同じです。
6. 剰余代入演算子(%=): 左辺の変数を右辺の値で割った余りを左辺の変数に代入します。例えば、**a %= 5;**は、**a = a % 5;**と同じです。

これらの代入演算子は、簡潔に変数に値を代入する際に便利です。また、加算代入演算子や減算代入演算子などは、変数の値を増減させる場合にも便利です。

以下にJavaの代入演算子のサンプルコードを示します。

int a = 10; // 代入演算子

a += 5; // 加算代入演算子

a -= 3; // 減算代入演算子

a \*= 2; // 乗算代入演算子

a /= 4; // 除算代入演算子

a %= 3; // 剰余代入演算子

**6.5 インクリメント/デクリメント演算子**

Javaにおいて、インクリメント演算子（++）とデクリメント演算子（--）は、数値型の変数に対して1を加算したり、減算したりするために使用されます。これらの演算子は単項演算子で、変数の前後に配置することができます。

例えば、以下のコードは、変数xの値を1増やすインクリメント演算子を使用しています。

int x = 10;

x++;

System.out.println(x); // Output: 11

同様に、以下のコードは、変数yの値を1減らすデクリメント演算子を使用しています。

int y = 5;

y--;

System.out.println(y); // Output: 4

Javaにおいて、インクリメント演算子（++）とデクリメント演算子（--）は、数値型の変数に対して1を加算したり、減算したりするために使用されます。これらの演算子は単項演算子で、変数の前後に配置することができます。

例えば、以下のコードは、変数xの値を1増やすインクリメント演算子を使用しています。

int x = 10; x++; System.out.println(x); // Output: 11

同様に、以下のコードは、変数yの値を1減らすデクリメント演算子を使用しています。

int y = 5; y--; System.out.println(y); // Output: 4

インクリメント演算子とデクリメント演算子は、forループやwhileループなどの反復処理でよく使用されます。また、インクリメント演算子とデクリメント演算子は、プログラムの可読性を向上させるために、単独で使用することが推奨されています。

6.6 ビット演算子

Javaには、ビット演算子と呼ばれるビット単位での演算を行うための演算子があります。以下に、主要なビット演算子を紹介します。

1)ビットAND演算子（&） ビットAND演算子は、対応する2つのビットの両方が1である場合に、結果のビットが1となります。それ以外の場合は、0となります。

例えば、以下のコードは、8ビットの2つの数値のビットAND演算を行います。

byte a = 0b00100110;

byte b = 0b00011010;

byte result = (byte) (a & b); // result = 0b00000010

2)ビットOR演算子（|） ビットOR演算子は、対応する2つのビットのいずれかが1である場合に、結果のビットが1となります。両方が0の場合には、0となります。

例えば、以下のコードは、8ビットの2つの数値のビットOR演算を行います。

byte a = 0b00100110;

byte b = 0b00011010;

byte result = (byte) (a | b); // result = 0b00111110

3)ビットXOR演算子（^） ビットXOR演算子は、対応する2つのビットが異なる場合に、結果のビットが1となります。両方が同じ場合には、0となります。

例えば、以下のコードは、8ビットの2つの数値のビットXOR演算を行います。

byte a = 0b00100110; byte b = 0b00011010; byte result = (byte) (a ^ b); // result = 0b00111100

4)ビット反転演算子（~） ビット反転演算子は、対象となるビットを反転させます。1は0に、0は1に変換されます。

例えば、以下のコードは、8ビットの数値のビット反転演算を行います。

byte a = 0b00100110;

byte result = (byte) (~a); // result = 0b11011001

ビット演算子は、ビット単位での演算を行うため、主にビットフィールドやマスク処理などのプログラムで使用されます。

**6.7 条件演算子（三項演算子）**

以下に、Javaの条件演算子（三項演算子）のサンプルコードを示します。

int a = 10;

int b = 20;

// aがbよりも大きい場合は"大きい"、そうでない場合は"小さい"という文字列を返す

String result = (a > b) ? "大きい" : "小さい";

System.out.println(result); // Output: "小さい"

上記のコードでは、条件演算子を使用して、変数**a**が変数**b**よりも大きい場合は文字列"大きい"を、そうでない場合は文字列"小さい"を返す処理を行っています。

**a**の値が**10**で**b**の値が**20**なので、条件**(a > b)**は**false**となり、条件演算子は式**"小さい"**を評価して返します。最終的に、"小さい"という文字列がコンソールに出力されます。

**6.8演算子の計算順番は？**

Javaにおいて、演算子の計算順序は、演算子の優先度と結合規則によって決まります。以下に、Javaの演算子の優先度を示します（優先度の高い順から低い順へ）。

1. 単項演算子（++, --, +, -, !, ~）
2. 乗算・除算・剰余演算子（\*, /, %）
3. 加算・減算演算子（+、-）
4. シフト演算子（<<, >>, >>>）
5. 比較演算子（<, >, <=, >=, instanceof）
6. 等価演算子（==, !=）
7. ビット演算子（&, ^, |）
8. 論理演算子（&&, ||）
9. 条件演算子（? :）
10. 代入演算子（=, +=, -=, \*=, /=, %=, <<=, >>=, >>>=, &=, ^=, |=）

優先度が同じ演算子は、左から右へ結合されます。ただし、代入演算子は右から左へ結合されます。

例えば、以下の式を見てみましょう。

int x = 10 + 20 \* 2;

この式では、**\***演算子の優先順位が高いため、**20 \* 2**が先に評価されます。その後、**10 + 40**が評価され、変数**x**には**50**が代入されます。

しかし、以下のようにかっこを使用することで、計算順序を明示的に指定することができます。

int x = (10 + 20) \* 2;

この場合、かっこの中が先に評価されるため、**10 + 20**が評価され、**30**となります。その後、**30 \* 2**が評価され、変数**x**には**60**が代入されます。