第3章　演算子

目次

- 演算子とは
- 算術演算子
- 単項演算子
- 代入演算子と複合代入演算子
- 関係演算子
- 論理演算子
- null比較

演算子とは

- Java言語で足し算や掛け算などの計算を行う場合に使用する

演算子	結合規則	優先順位
(引数) [配列添字] . ++ --(後置き)	左→右	高
! ~ + - (単項演算子) ++ --(前置き)	右→左	
new (型)		
* / %	左→右	
+ - (算術演算子)		
<< >> >>>		
> >= < <= instanceof		
== !=		
&		
^		
|		
&&		
||		
?:	右→左	
= ope=		低

※ ope=は「+=」「-=」「*=」「/=」「%=」「<<=」「>>=」「>>>=」「&=」「|=」「^=」のこと

算術演算子

演算子	機能	書式	意　　味
＋	加算	a＋b	aとbを加える
－	減算	a－b	aからbを引く
＊	乗算	a＊b	aとbを掛ける
／	除算	a/b	aをbで割る
％	剰余	a％b	aをbで割った余りを求める

〔注〕　剰余を求める場合は、aとbは整数である必要がある。

演算子の使用例

```
4  public class Main {
5      public static void main(String[] args)  {
6          System.out.println(10/3.0);
7
8          System.out.println(10 % 3);
9      }
10 }
11
```

実行 (Ctrl-Enter)

出力　入力　コメント 0

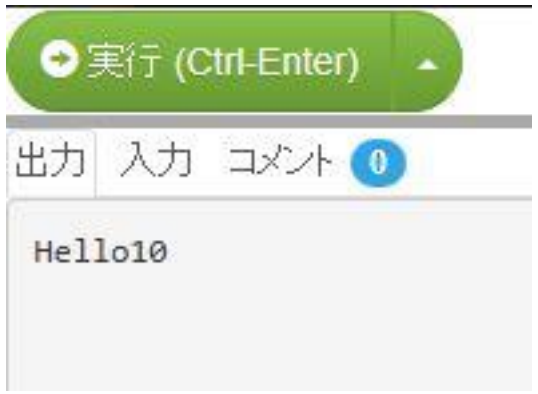
```
3.3333333333333335
1
```

・整数/浮動小数点数の結果は浮動小数点数となる

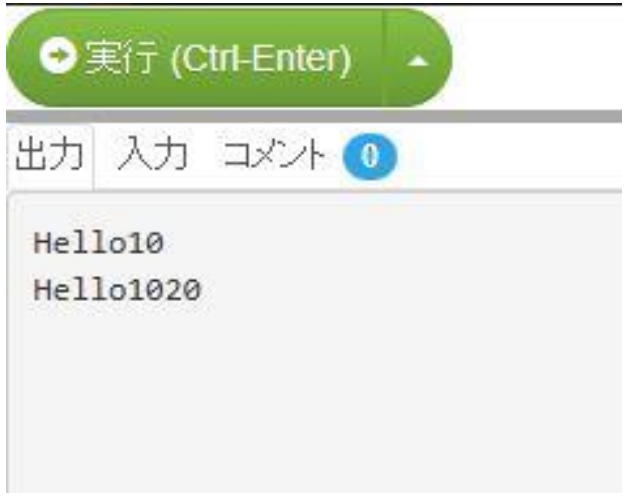
・%は余りを求める演算子なので「１」となる

演算子による数字と文字列の処理

```
4  public class Main {
5      public static void main(String[] args)  {
6          String str = "Hello";
7          int a = 10;
8          int b = 20;
9
10         System.out.println(str + a);
11         //System.out.println(str + a + b);
12     }
13 }
14
```



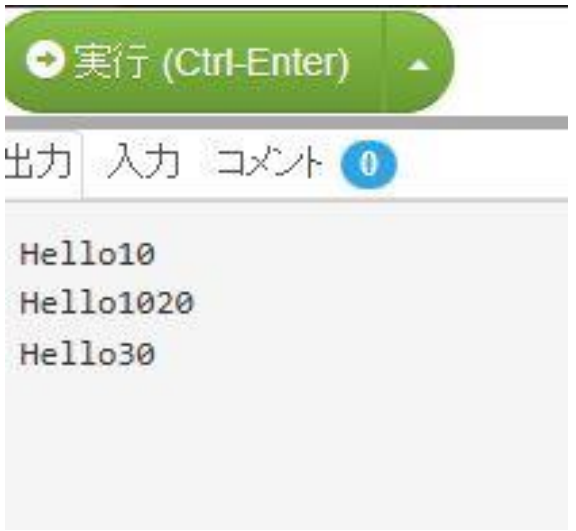

```
System.out.println(str + a);
System.out.println(str + a + b);
```



演算子による数字と文字列の処理（2）

```
System.out.println(str + a);
System.out.println(str + a + b);

System.out.println(str + (a + b));
```



・演算処理は左から行われる

・（）でくくったところから、先に計算される

単項演算子

演算子	記述例	説明
－	－a	aの符号を反転させる
＋＋	＋＋a、a＋＋	aの値に1を加える
－－	－－a、a－－	aの値から1を引く

前置（演算子を変数の前に配置）

```
int a = 10;
Int b = ++a;
```

①計算してから
②代入する

後置（演算子を変数の後に配置）

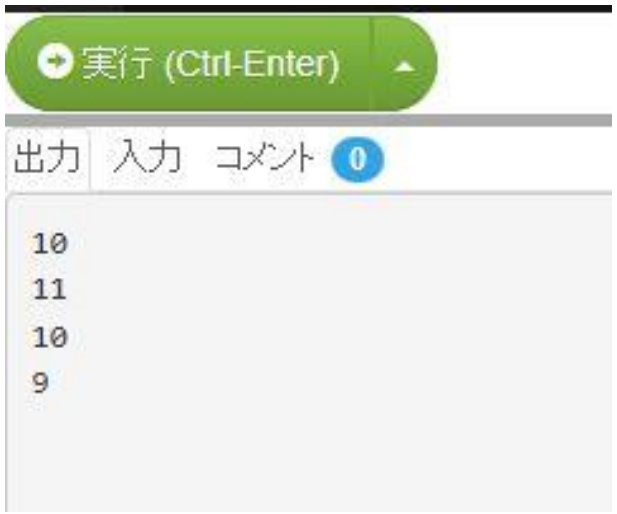
```
int a = 10;
Int b = a++;
```

①代入してから
②計算する

単項演算子（2）

＋＋演算子と－－演算子の動作

```
4  public class Main {
5      public static void main(String[] args)  {
6          int a =10; int b=10; int c =10; int d =10;
7          System.out.println(a++);
8          System.out.println(++b);
9          System.out.println(c--);
10         System.out.println(--d);
11     }
12 }
13
```



・aとcについては、変数値の出力がされてから、インクリメント/デクリメントが行われた

・bとdについては、インクリメント/デクリメントが行われてから、変数値の出力がされた

代入演算子と複合代入演算子

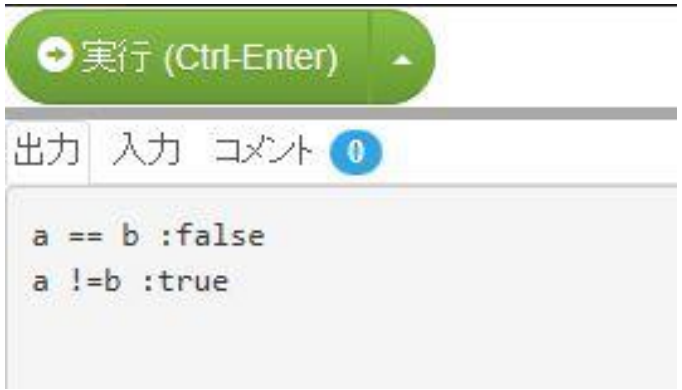
演算子	記述例	説明	算術演算子での記述
＝	a=b	aにbを代入する	
＋＝	a+=b	aにbを加えた値をaに代入する	a=a+b
－＝	a-=b	aからbを引いた値にaを代入する	a=a-b
*＝	a*=b	aにbを乗した値をaに代入する	a=a*b
/＝	a/=b	aをbで割った値をaに代入する	a=a/b
%＝	a%=b	aをbで割った余りをaに代入する	a=a%b

関係演算子

演算子	記述例	説明
＝＝	a==b	aとbの値が等しければtrue、異なればfalse
！＝	a!=b	aとbの値が異なればtrue、等しければfalse
＞	a>=b	aの値がbより大きいならばtrue、以下ならばfalse
＞＝	a>=b	aの値がb以上であればtrue、小さければfalse
＜	a<b	aの値がbの値より小さければtrue、以上ならばfalse
＜＝	a<=b	aの値がbの値以下であればtrue、大きければfalse

関係演算子(2)

```
4  public class Main {
5      public static void main(String[] args)  {
6          // Here your code !
7          int a =10; int b =20; int c =10;
8          System.out.println("a == b :"+(a == b));
9          System.out.println("a !=b :"+( a != b));
10
11     }
12 }
13
```



・両辺の値が等しいかの比較には＝＝演算子を使用する

論理演算子

演算子	記述例	説明
&	a&b	aとbの両方がtrueのときtrue、そうでなければfalse aがfalseであったとしてもbは評価される
& &	a&&b	aとbの両方がtrueのときtrue、そうでなければfalse aがfalseならbは評価されず結果がfalseとなる aがtrueならbも評価され結果を返す
|	a|b	aとbいずれかがtrueならtrue、そうでなければfalse aがtrueであったとしてもbは評価される
| |	a||b	aとbいずれかがtrueならtrue、そうでなければfalse aがtrueならbは評価されず結果がtrueとなる aがfalseならbも評価され結果を返す
^	a^b	aとbの値が異なるとき、true、そうでなければfalse
!	!a	aの値がtrueのとき、false、falseのときtrue

論理演算子（2）

and条件：2つの条件がともにtrueならば、trueを返す

条件1＆条件2

条件1がfalseでも条件2を必ず評価する

条件1＆＆条件2

条件1がfalseなら条件2は評価しない

or条件：条件のいずれかがtrueならば、trueを返す

条件1 | 条件2

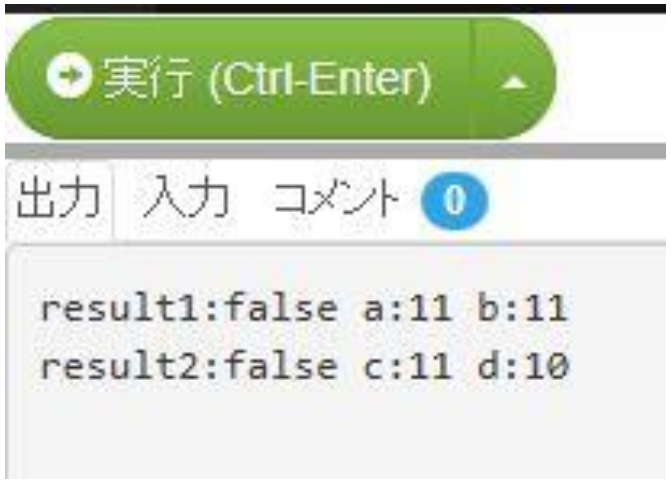
条件1がtrueでも条件2を必ず評価する

条件1 || 条件2

条件1がtrueなら条件2は評価しない

論理演算子(3)

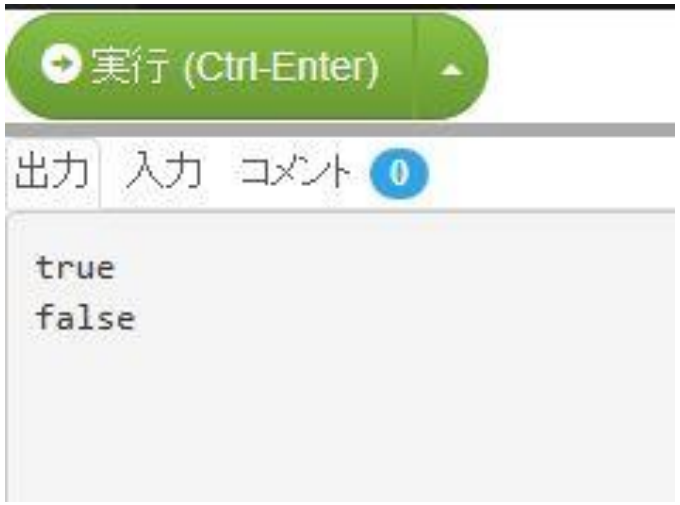
```
4  public class Main {
5      public static void main(String[] args)  {
6          // Here your code !
7          int a=10; int b=10; int c=10;int d=10;
8          boolean result1=a++ > 10 & ++b > 10;
9          System.out.println("result1:" + result1 + " a:" + a + " b:"+ b);
10         boolean result2=c++ > 10 && ++d >10;
11         System.out.println("result2:" + result2 + " c:" + c + " d:"+ d);
12     }
13 }
14
```



組み合わせ	&、&&	|、||
true-true	true	true
true-false	false	true
false-true	false	true
false-false	false	false

null比較

```
4  public class Main {
5      public static void main(String[] args)  {
6          String str1 = null;
7          String str2 ="";
8
9          System.out.println(str1==null);
10         System.out.println(str2==null);
11     }
12 }
13
```



・空文字とnullとは異なり、メモリ領域は確保されている。