Ｊａｖａ入門

(№.２)

科　名		氏　名	
Tutorial group			

2004.10

1 式(expression)と演算子(operator)

Ｊａｖａでは、次のものを**式**(expression)として取り扱う。

① リテラル(literal)、変数(variable)

② メソッド(method)の呼び出し、配列要素へのアクセス

③ ①～②の各**式**を演算子で結合したもの

④ インスタンス(instance)の生成、配列生成

なお、**式**に末尾に";"を付加したものを**式文**と言う。　(The Java Language Specification)

次に、Ｊａｖａの演算子(operator)を示す。

① 算術演算子(+, -, *, /, %, ++, --)

② 比較演算子(<, <=, >, >=, ==, !=)

③ 論理演算子(&&, &, ||, |, !)

④ ビット演算子(&, |, ^, ~)

⑤ シフト演算子(<<, >>, >>>)

⑥ 代入演算子(=, +=, -=, *=, /=, %=, &=, |=, ^=, <<=, >>=, >>>=)

⑦ 文字列連結演算子(+)

⑧ 条件演算子(? :)

⑨ instanceof 演算子(instanceof)

【例題 1】 ― *演算子, +演算子

[ファイル名]

0papp1.java

[プログラムの説明]

テキスト・P114 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp1"とすること。

【考察】

プログラム中の演算子(*演算子、+演算子)を示し、説明しなさい。

.....

.....

.....

.....

【例題 2】 ― 演算子の優先順位

[ファイル名]

0papp2.java, 0papp2a.java

［プログラムの説明］

テキスト・P119 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp2"とすること。

［プログラムの説明］

テキスト・P120 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp2a"とすること。

【考察】

【例題 3】－ ++演算子

［ファイル名］

0papp3.java

［プログラムの説明］

テキスト・P121 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp3"とすること。

【考察】

【例題 4】－ ~演算子, !演算子

［ファイル名］

0papp4.java

［プログラムの説明］

テキスト・P123 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp4"とすること。

◆ ~演算子 … 各ビットを、ビット毎に反転した結果を返す。

◆ !演算子 … オペランドの論理否定を返す。

【考察】

【例題 5】 ― *演算子, /演算子

［ファイル名］

0papp5.java

［プログラムの説明］

テキスト・P124 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp5"とすること。

【考察】

【例題 6】 ― +演算子, -演算子

［ファイル名］

0papp6.java

［プログラムの説明］

テキスト・P125 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp6"とすること。

【考察】

【例題 7】 ― <<演算子, >>演算子, >>>演算子

［ファイル名］

0papp7.java

［プログラムの説明］

テキスト・P126 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp7"とすること。

◆ <<演算子 … 整数型の式を指定されたビット数分、左向きに**論理シフト**する。

◆ >>演算子 … 整数型の式を指定されたビット数分、右向きに**算術シフト**する。

◆ >>>演算子 … 整数型の式を指定されたビット数分、右向きに**論理シフト**する。

（**算術シフト**は符号ビットを保証するが、**論理シフト**は符号ビットを保証しない。）

【考察】

【例題 8】－ <演算子

［ファイル名］

0papp8.java

［プログラムの説明］

テキスト・P127 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp8"とすること。

【考察】

【例題 9】－ &演算子, <<演算子

［ファイル名］

0papp9.java

［プログラムの説明］

テキスト・P128 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp9"とすること。

【考察】

【例題 10】－ <<演算子, |演算子, ==演算子

［ファイル名］

0papp10.java

［プログラムの説明］

テキスト・P129 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp10"とすること。

【考察】

【例題 11】－ <演算子, >演算子, &演算子

［ファイル名］

0papp11.java

［プログラムの説明］

テキスト・P130 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp11"とすること。

【考察】

【例題 12】－ <演算子, >演算子, &&演算子

［ファイル名］

0papp12.java

［プログラムの説明］

テキスト・P131 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp12"とすること。

◆ &&演算子と||演算子は**短絡演算子**(short-circuit operator)と呼ばれ、左辺のオペランドの評価によってその演算に必要な情報をすべて判断できた場合、右辺のオペランドの評価を行わない。

一方、&演算子と|演算子は、左辺のオペランドの評価に関係なく右辺のオペランドの評価を行う。

【考察】

"0papp11.java"との違いを考察しなさい。

【例題 13】－ !=演算子, <演算子, &&演算子

［ファイル名］

0papp13.java

［プログラムの説明］

テキスト・P132 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp13"とすること。

【考察】

【例題 14】－ &&演算子, &演算子

［ファイル名］

0papp14.java

［プログラムの説明］

テキスト・P132 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp14"とすること。

【考察】

【例題 15】－ ? :演算子

［ファイル名］

0papp15.java

［プログラムの説明］

テキスト・P133 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp15"とすること。

【考察】

【例題 16】－ =演算子

［ファイル名］

0papp16.java

［プログラムの説明］

テキスト・P134 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp16"とすること。

【考察】

【例題 17】－ =演算子

［ファイル名］

0papp17.java

［プログラムの説明］

テキスト・P135 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp17"とすること。

【考察】

【例題 18】－ *=演算子

［ファイル名］

0papp18.java

［プログラムの説明］

テキスト・P135 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp18"とすること。

【考察】

【例題 19】－ Math クラスのメソッド

［ファイル名］

0papp19.java

［プログラムの説明］

テキスト・P136 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp19"とすること。

【考察】

【例題 20】－ String クラスのメソッド

［ファイル名］

0papp20.java

［プログラムの説明］

テキスト・P138 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"0papp20"とすること。

【考察】

【例題 21】

［ファイル名］

Opapp21.java

［プログラム］

```
public  class   Opapp21 {
    public  static  void    main(String args[]) {   //  Ｊａｖａエントリポイント
        //  MovePhone クラスオブジェクトの生成
        MovePhone mp = new  MovePhone("0120-123-456");
        instanceofTest(mp);

        //  FixedPhone クラスオブジェクトの生成
        FixedPhone fp = new FixedPhone("050-1234-567", "Aichi Pref.");
        instanceofTest(fp);
    }
    //  instanceof 演算子のテスト
    public  static  void    instanceofTest(Telephone tel)   {
        if(tel instanceof Telephone){               //  スーパークラスとの比較
            System.out.println("TEL is Telephone object.");
        }
        if(tel instanceof MovePhone){               //  MovePhone クラスとの比較
            System.out.println("TEL is Movephone object.");
        }
        if(tel instanceof FixedPhone){              //  FixedPhone クラスとの比較
            System.out.println("TEL is Fixedphone object.");
        }
        System.out.println("--------------------------");
    }
}   //  End of Opapp21

class   Telephone   {                       //  スーパークラスの定義
    //  インスタンス変数
    String  TelephoneNumber;                //  電話番号
}
class   MovePhone   extends Telephone   {   //  サブクラス(MovePhone)の定義
    //  追加するインスタンス変数
    int BatteryRemaining = 100;             //  電池残量

    //  コンストラクタ
    public  MovePhone(String TelNum){
        TelephoneNumber = TelNum;
    }
```

```
}
class   FixedPhone  extends Telephone   {   //  サブクラス(FixedPhone)の定義
    //  追加するインスタンス変数
    String  InstallationSite;               //  地域名

    //  コンストラクタ
    public  FixedPhone(String TelNum, String Site){
        TelephoneNumber = TelNum;
        InstallationSite = Site;
    }
}
```

◆ **継承**(inheritance)

クラスに共通する「属性」・「振る舞い」を括ってより汎用的なクラスを作成することを「**汎化**」と言う。⇒ **スーパークラス**(super class)

そして、個々のクラスでは、この汎用的なクラスの共通する内容をそのまま利用して異なる部分だけを定義する。これを「**特化**」と言う。⇒ **サブクラス**(sub class)

このように、あるクラスの共通内容を引き継ぐ仕組みを「**継承**」と言う。

すなわち、**サブクラス**では**スーパークラス**との「**差分**」だけを定義することになる。

《サブクラスで定義する事項》
- **サブクラス**だけが持つ「属性」・「振る舞い」
- **スーパークラス**の「振る舞い」に対する変更

【考察】

2 if 文

制御文は、**文**(statement)の実行順序を制御する。

if 文は、**条件式**(condition)を評価して実行する**文**を選択する。

【例題 22】 ― if 文(if-then 文)

[ファイル名]

Ifapp1.java, Ifapp1a.java

[プログラムの説明]

テキスト・P140 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Ifapp1"とクラス名"Ifapp1a"すること。

※ 実行する**文**が複数ある場合は、**{}**で囲む必要がある。⇒ 複文(block)

【考察】

【例題 23】－ if-then-else 文

［ファイル名］

Ifapp2.java

［プログラムの説明］

テキスト・P141 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Ifapp2"とすること。

【考察】

【例題 24】

［ファイル名］

Ifapp3.java

［プログラムの説明］

テキスト・P142 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Ifapp3"とすること。

【考察】

【例題 25】－ if-then-else if-else 文

［ファイル名］

Ifapp4.java

［プログラムの説明］

テキスト・P142 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Ifapp4"とすること。

【考察】

3 switch 文

switch 文は、**式**の値により実行する**文**を選択する。
式は、byte 型、short 型、int 型及び char 型のいずれかの値を返すものであること。

【例題 26】－ switch 文
［ファイル名］
Swapp1.java
［プログラムの説明］
テキスト・P144 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Swapp1"とすること。
【考察】

【例題 27】
［ファイル名］
Swapp2.java
［プログラムの説明］
テキスト・P145 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Swapp2"とすること。
【考察】

4 while 文

while 文は、**条件式**を評価して**条件式**が true(**真**)の間、**文**を繰返し実行する。
while 文は、**文**の実行前に**条件式**を評価するため、**文**が 1 回も実行されない場合もある。

【例題 28】－ while 文
［ファイル名］
Whapp1.java
［プログラムの説明］
テキスト・P146 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Whapp1"とすること。

【考察】

【例題 29】

［ファイル名］

Whapp2.java

［プログラムの説明］

テキスト・P147 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Whapp2"とすること。

【考察】

【例題 30】

［ファイル名］

Whapp3.java

［プログラムの説明］

テキスト・P148 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Whapp3"とすること。

【考察】

【例題 31】

［ファイル名］

Whapp4.java

［プログラムの説明］

テキスト・P148 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Whapp4"とすること。

【考察】

5 do-while 文

do-while 文は、**条件式**を評価して**条件式**が true(**真**)の間、**文**を繰返し実行する。
do-while 文は、**文**の実行後に**条件式**を評価するため、**文**が最低１回実行される。

【例題 32】－ do-while 文

［ファイル名］
Doapp1.java
［プログラムの説明］
テキスト・P149 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Doapp1"とすること。
【考察】

【例題 33】

［ファイル名］
Doapp2.java, Doapp2a.java
［プログラムの説明］
テキスト・P150 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Doapp2"とクラス名"Doapp2a"とすること。
【考察】

6 for 文

for 文は、**条件式**を評価して**条件式**が true(**真**)の間、**文**を繰返し実行する。
for 文は、繰返し実行する回数があらかじめ分かっている場合に適している。
for 文は、**初期化**を実行後、**文**の実行前に**条件式**の評価をするため、**文**が１回も実行されない場合もある。

【例題 34】－ for 文

［ファイル名］
Forapp1.java
［プログラムの説明］
テキスト・P151 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名

"Forapp1"とすること。

【考察】

【例題 35】

［ファイル名］

Forapp2.java

［プログラムの説明］

テキスト・P152 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Forapp2"とすること。

【考察】

【例題 36】

［ファイル名］

Forapp3.java

［プログラムの説明］

テキスト・P153 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Forapp3"とすること。

【考察】

【例題 37】

［ファイル名］

Forapp4.java

［プログラムの説明］

テキスト・P153 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Forapp4"とすること。

【考察】

【例題 38】

［ファイル名］

Forapp5.java

［プログラムの説明］

テキスト・P154 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Forapp5"とすること。

【考察】

【例題 39】

［ファイル名］

Forapp6.java

［プログラムの説明］

テキスト・P155 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Forapp6"とすること。

【考察】

7 break 文

break 文は、switch 文、while 文、do-while 文及び for 文の処理を中止し、次の**文**へ制御を移す。

【構文①】

break　［ラベル］;

- ラベルは省略可能。
- break 文(ラベル無し)によって処理を中止することができる**文**は、その break 文を含む最も内側の switch 文、while 文、do-while 文及び for 文である。
- ラベル付き break 文によって、ネスト(nest)した繰返し**文**を終了することができる。

【例題 40】－ break 文

［ファイル名］

Forapp7.java

［プログラムの説明］

テキスト・P156 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Forapp7"とすること。

【考察】

8 continue 文

continue 文は、while 文、do-while 文及び for 文の処理を省略し、次の繰返し処理を開始する。

【構文②】

continue　[ラベル];

- ラベルは省略可能。
- continue 文(ラベル無し)によって処理を省略することができる**文**は、その continue 文を含む最も内側の while 文、do-while 文及び for 文である。
- ラベル付き continue 文によって、任意のブロックから処理を再開することができる。

【例題 41】

［ファイル名］

Forapp8.java

［プログラムの説明］

テキスト・P157 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Forapp8"とすること。

【考察】

【例題 42】

［ファイル名］

Forapp9.java

［プログラムの説明］

テキスト・P158 app クラスのプログラムを作成しなさい。ただし、クラス名"app"はクラス名"Forapp9"とすること。

【考察】