変数と演算子

変数

プログラムでデータを取り扱う場合には対象となるデータを保存する必要がでてくる。このデータを保存する場所のことを「変数」と呼び、変数を説明する上ではよく「データを入れておく箱」として例えられ、必要に応じて中身（データ）は後から自由に変更することが可能である。詳細については次回以降で解説するが、今回は標準出力と絡めて、多少説明を行っていく。

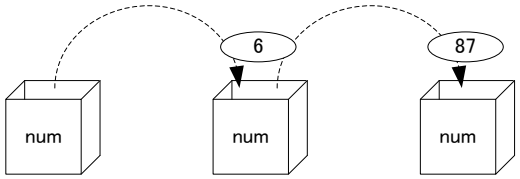

図１　変数

変数は以下のように宣言を行うことで利用する。

書式
データ型　変数名;

データ型にはいくつかあるが今回は以下の２つを紹介する。

int　　整数用のデータ型
double　実数用のデータ型

変数名は識別子のところで示したルールに従い、各自で自由に設定できる。ただし、変数名については全て小文字を用いる習慣がある。また、変数は以下のようにして、データを保存することで利用する。この変数にデータ（値）を保存する処理のことを「変数に値を代入する」という。

書式
変数名 = 値;

たとえば、num という変数名の int 型の変数に「100」を代入するには、以下の使用例のように行う。

使用例
int num; num = 100;

また、変数の宣言と値の代入は以下のように同時に行うことができる。

使用例
int num = 100;

「変数の宣言と同時に値の代入を行う」場合を特に「初期化」と言い、その時の値を「初期値」という。ただし、より一般的には初めて変数に値を代入する場合も含めて「初期化」と言っていると思われるので注意が必要である。

変数に代入した値の表示

以下はint型の整数の変数aを宣言して値を代入した後にその値を表示し、さらに文字「a」そのものを表示するプログラムである。

プログラム例１

```
public class Prog02_01
{
        public static void main(String[ ] args)
        {
                int a = 10;

                System.out.println(a);
                System.out.println("a");
        }
}
```

出力例

```
10
a
```

変数に保存した値を出力する際には、直接変数名を記述する。また、ダブルクォーテーション（"）で囲んだ部分は文字列として出力される。

変数を利用したプログラムでは以下のように代入や四則演算といった処理を行うことが可能である。

プログラム例２

```
public class Prog02_02
{
        public static void main(String[ ] args)
        {
                int a;
                double b;

                a = 56 + 71;
                System.out.println(a);

                b = a;
                System.out.println(b);

                //a = b;
                //System.out.println(a);
```

```
        }
}
```

出力例

```
127
127.0
```

補足

コメントアウトをはずしたものも試すこと。

演習としてプログラム例を作成して実行を確認してもらうことになるが、プログラム例６についてはコメントアウトをはずしたもの（//を削除する）も試してもらいたい。この場合、コンパイルエラーが発生することが確認できる。原因は int 型である整数型変数に double 型である実数型変数の値を代入しようとしていることにある。この場合、小数点以下の値を捨ててしまうことになるため、コンパイルエラーとして検出される。プログラミングではこのような処理を行いたいことも頻繁にある。その際には以下のように対応すればよい。

```
a = (int)b;
```

この処理はキャストといい、代入している値が整数型で正しいことをコンパイラに知らせている。

計算式と表示

System.out.println や System.out.print では以下の例のように計算式を含めて表示を行うことができる。

プログラム例３

```
public class Prog02_03
{
        public static void main(String[ ] args)
        {
                System.out.println( "45 + 29 = 74" );
                System.out.println( "45 + 29 = " + (45 + 29) );
        }
}
```

出力例

```
45 + 29 = 74
45 + 29 = 74
```

出力例では２行とも同じであるが、処理の内容が異なる。一行目は"45 + 29 = 74"という文字列を表示しており、二行目では"45 + 29 = "という文字列に(45 + 29)という計算を行った結果の 74 という数値を文字列"74"に変換してから、連結を行って表示をしている。

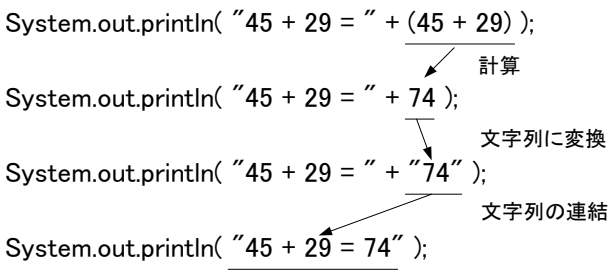

図２　処理の流れ

ここで注意しなくてはならないのは、(45＋29)の括弧は必ず付けなくてはならない。また、+は処理の順番によって文字列の連結になったり、数値の和を求めたりすることになるので注意が必要である。以下の出力例に注意すること。

プログラム例４

```
public class Prog02_04
{
        public static void main(String[ ] args)
        {
                System.out.println( "45 + 29 = 74" );
                System.out.println( "45 + 29 = " + 45 + 29);
                System.out.println(45 + 29 + "は 45 と 29 の和である");
                System.out.println((45 + 29) + "は 45 と 29 の和である");
        }
}
```

出力例

```
45 + 29 = 74
45 + 29 = 4529
74 は 45 と 29 の和である
74 は 45 と 29 の和である
```

出力例の三行目については以下の図に従った処理が行われている。

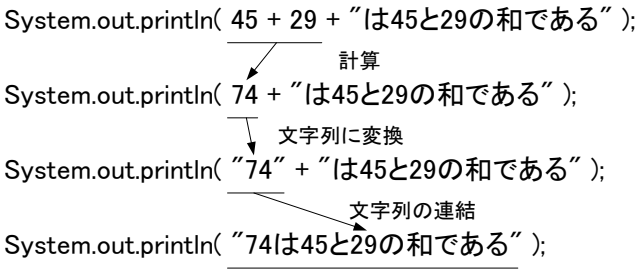

図３　処理の流れ

この例の場合には先に数値の和が計算される。その後、数値の「74」が文字列に変換され、最後に文字列

が連結されて出力される。図３の処理の流れとよく比較しておくこと。

計算結果については以下のように実数での表示も可能である。

プログラム例５

```
public class Prog02_05
{
        public static void main(String[ ] args)
        {
                System.out.println( "10 / 3 = " + (10 / 3) );
                System.out.println( "10.0 / 3.0 = " + (10.0 / 3.0) );
                System.out.println( "1 + 2 = " + (1 + 2) );
                System.out.println( "1.0 + 2.0 = " + (1.0 + 2.0) );
                System.out.println( "0.5 + 0.125 = " + (0.5 + 0.125) );
        }
}
```

出力例

```
10 / 3 = 3
10.0 / 3.0 = 3.3333333333333335
1 + 2 = 3
1.0 + 2.0 = 3.0
0.5 + 0.125 = 0.625
```

10.0/3.0 の計算結果は割り切れない。出力例では表示は小数点以下 16 桁であり、最後の桁については切り捨てや四捨五入といった形にはなっていない。

プログラム例５について変数を利用して書き直したプログラム例を以下に示す。

プログラム例６

```
public class Prog02_06
{
        public static void main(String[ ] args)
        {
                int a, b, c;
                double d, e, f;

                a = 10;
                b = 3;
                c  = a / b;
                System.out.println( "10 / 3 = " + c );

                d = 10;
                e = 3;
                f  = d / e;
                System.out.println( "10.0 / 3.0 = " + f );
```

```
                a = 1;
                b = 2;
                c  = a + b;
                System.out.println( "1 + 2 = " + c );

                d = 1;
                e = 2;
                f  = d + e;
                System.out.println( "1.0 + 2.0 = " + f );

                d = 0.5;
                e = 0.125;
                f  = d + e;
                System.out.println( "0.5 + 0.125 = " + f );
        }
}
```

出力例

```
10 / 3 = 3
10.0 / 3.0 = 3.3333333333333335
1 + 2 = 3
1.0 + 2.0 = 3.0
0.5 + 0.125 = 0.625
```

書式付き出力

Java は比較的頻繁にバージョンアップを重ねている。J2SE 5.0（内部的には 1.5）以降のバージョンではデータの出力を C 言語の printf のように指定できる書式付き出力が導入された。以下に使用例を示す。

プログラム例 7

```
public class Prog02_07
{
        public static void main(String[ ] args)
        {
                int a, b, c, d, e;
                double f;

                a = 1;
                b = 10;
                c = 100;
                d = 1000;
                e = 10000;
                f = 10.0/3.0;
```

```
                System.out.printf("書式付き出力の例¥n");
                System.out.printf("そのまま出力¥n");
                System.out.printf("%d¥n%d¥n%d¥n%d¥n%d¥n", a, b, c, d, e);
                System.out.printf("%f¥n", f);

                System.out.printf("桁数を調整して出力¥n");
                System.out.printf("%5d¥n%5d¥n%5d¥n%5d¥n%5d¥n", a, b, c, d, e);
                System.out.printf("%6.3f¥n", f);
        }
}
```

出力例

```
書式付き出力の例
そのまま出力
1
10
100
1000
10000
3.333333
桁数を調整して出力
    1
   10
  100
 1000
10000
 3.333
```

printf はダブルクォーテーション（"）で囲むことで文字列を表示することが可能である。改行には拡張表記の「¥n」を用いる。%○は変換仕様と呼ばれ、「%d」で整数値の表示、「%f」で実数値の表示に対応する。これらを用いる際には以下のようにして、文字列リテラルの後をカンマで区切り、変換仕様の個数と種類に対応した数値、変数、式を記述する。

```
        System.out.printf("%f¥n", f);
```

また、変換仕様には桁数を調整することも可能である。例えば「%3d」とすると 3 桁の整数となり、表示する数値の桁数が少ない場合にはスペースを表示することで調整する。また、「%f」の場合には自動的に小数点以下の桁数を 6 桁で表示するが、「%.4f」とすると小数点以下の桁数を 4 桁に調整する。実数の全体の桁数を調整するには「%9.6f」とする。この場合は小数点以下の桁数を 6 桁、全体の桁数を 9 桁で調整する。この場合には小数点「.」は 1 桁扱いになるため、整数部分の表示は 2 桁に調整されるので注意が必要である。

演習

・テキストで示したプログラム例１～７を作成すること。
・値を適宜変更するなど、いろいろ試すこと。

・下図のような実行結果を表示させるプログラムを作成しなさい。

（注意）計算結果は計算式で記述すること。表示については指定する桁数も考慮すること。

□部分は全角のスペースである。■部分は半角のスペースである。■の表示については桁数の調整で表示すること

は	じ	め	て	の	□	J	A	V	A
3 + 2 = 5									
9 + 7 = ■ 1 6									
1 0 / 3 = ■ ■ 3 . 3 3 3 3									