5　プログラミングの習慣

ここまで多くのプログラムを書いてきましたが、そろそろプログラムの行数も多くなり、自分が書いてきたプログラムを自分で読み返しても理解しにくい、という状況になり始めていることと思います。

プログラミングには、プログラムを後から自分で読み返したとき、あるいはプログラムを他の人に渡したとき、などのために便利な習慣や規則がいくつか伝承されています。ここで、世界中の多くのプログラマが倣っている、プログラミングのいくつかの習慣や規則について紹介します。

5.1　インデント

いままでに紹介したプログラムでは、命令や構文 1 個ごとに改行をしましたし、また特定の行では行頭に所定の空白文字を加えていました。

しかし実をいうと、Java 言語や C 言語を含むいくつかのプログラム言語では、改行や行頭空白文字はプログラムに影響を及ぼしません。

極端な例として、行頭空白文字を全部消したプログラムを書いてみます。

```
public class Hello {
public static void main(String[] args) {
System.out.println("Hello, world!");
}
}
```

これで javac コマンドと java コマンドを使ってみてください。どうですか？ 普通に Hello, world ! と表示されますよね？

今度はさらに、改行も全部消してみましょう。（本書では紙面の大きさ都合により、途中で折り返されて表示されるかもしれません。）

```
public class Hello { public static void main(String[] args) { System.out.println("Hello,
world!"); } }
```

これで javac コマンドと java コマンドを使ってみてください。どうですか？ 普通に Hello, world ! と表示されますよね？

しかし実際には、ほとんどのプログラマは、今までに本書で出てきたのと同様に、プログラムに改行や行頭空白文字を加えています。それは、ほとんどのプログラマが、この書き方が「読みやすい」と考えているからであり、また「世界中のプログラマが同じようなマナーでプログラムを書いたほうが便利」と考えているからです。

今後のプログラムの書き方の目安として、

- 命令や代入文 1 個ごとに改行する。
- 中カッコを開く（{）たびに、行頭空白文字の数を増やす。
- 中カッコを閉じる（}）たびに、行頭空白文字の数を元に戻す。

ということを習慣化してください。

この行頭空白文字を **インデント**または **字下げ** と呼ぶことがあります。

なお、ここで気をつけて欲しいのは、

日本語入力時の全角空白文字を、インデントに用いることはできない

という点です。くれぐれも忘れないで下さい。

5.2 カッコの省略

if 文、for 文、while 文、do-while 文などに付随して実行される処理を囲む中カッコ（{ と }）は、カッコ内の処理が 1 文（セミコロン 1 個）だけのときに省略することができます。例えば

```
if(a >= 1) {
   System.out.println(a);
}
```

は

```
if(a >= 1)
   System.out.println(a);
```

というように書き換えることができます。ただし本書では、このようなカッコの省略は行いません。

5.3 コメント

プログラムを構成する各々の処理や命令の意味をわかりやすくするために、多くのプログラマは**コメント**を記入します。

コメントは計算機がコンパイル時に読み飛ばすものであり、プログラマを含む人間が画面などの上で読むだけのために記載されます。

コメントの書き方は、以下の 2 種類です。

- /* と */ で囲まれた間にコメントを書く。この間に改行があってもよい。
- // から後、その行だけにコメントを書く。

以下に、コメントや空行を意図的に多く加えたプログラムの例を書きます。本書では以下、日本語でコメントを記入することにします。

```
// BMI 値を計算するクラス
public class BmiMethod {

   // 体重と身長を受け取り、BMI 値を算出して返すメソッド
   public static double calculateBmi(double weight, double height) {
     // BMI の公式は (体重)/(身長^2)
     double result = weight / (height * height);
     return result;
   }

   // メインメソッド
```

```
    public static void main(String[] args) {

        double w = 67.0; // 体重 67.0kg
        double h = 1.78; // 身長 1.78m

        // BMI 値を算出して、変数 bmi に代入する
        double bmi = calculateBmi(w, h);

        // BMI 値を画面に出力する
        System.out.println("BMI=" + bmi);
    }
}
```

5.4 変数名・定数名・クラス名・メソッド名

プログラムに出てくる変数、定数、クラス、メソッドなどの名前にも一定の習慣があり、多くのプログラマがそれに倣ってプログラムを書いています。本書では以下の習慣によって、変数、定数、クラス、メソッドの名前をつけています。参考になれば幸いです。

変数名 小文字だけで記述する。

(例) k, m, n, count, key, light

定数名 大文字だけで記述する。2 つ以上の単語を合成した定数名をつくる場合には、単語と単語の間をアンダースコアで接続する。

(例) COUNT, KEY, ONE_TIME, COLOR_RED

メソッド名 原則として小文字だけで記述する。ただし複数の単語を組み合わせて名前をつくる場合には、最初の単語は小文字だけで記述し、2 つめ以降の単語は最初の文字だけを大文字にする。

(例) get(), set(), initialize(), getAnswer(), setValue() なお 2 つ以上の単語を用いるときの習慣として、「動詞＋目的語」という単語の組み合わせが非常に多い。

クラス名 最初の文字だけを大文字にする。複数の単語を組み合わせて名前をつくる場合には、各単語の最初の文字だけを大文字にする。

(例) Color, Shape, RedTriangle, FirstGrade

そのクラスが一定の処理を担当するときには、「…の処理をする人」という意味をこめて -er で終わる接尾語を使うことが多い。

(例) Calculator, ColorInitializer, MessageReceiver