（2008-05-20 版）

Java によるプログラミング入門 (3)

1　Lesson 1: 代入式と変数

1.1　タスク 1

先ほどのプログラムは，柱状のタンクに水を入れていったときのタンクの水位を求めるプログラムです．画面に先ほど打ち込んだプログラムを表示させ，見比べながら話を聞いてください．

そこには以下のような量が現れました．() 内が「単位」です．

- タンクの底面積 (m^2).
- 水を入れる前の水位 (初期水位，m).
- 水の毎秒当たりの流入量 (m^3/s).
- 水位を求めたい時刻（水を入れ始めてからの経過時間，s).

このとき，タンクの水位は

$$\text{水位} = \text{初期水位} + \text{毎秒の流入量} \times \text{経過時間}／\text{底面積} \qquad (1)$$

で表されます．これを求めるプログラムが先ほどの TankCalculator.java です．

単純なプログラムですが, ここにはとても多くの, 基本的な概念が登場しています．

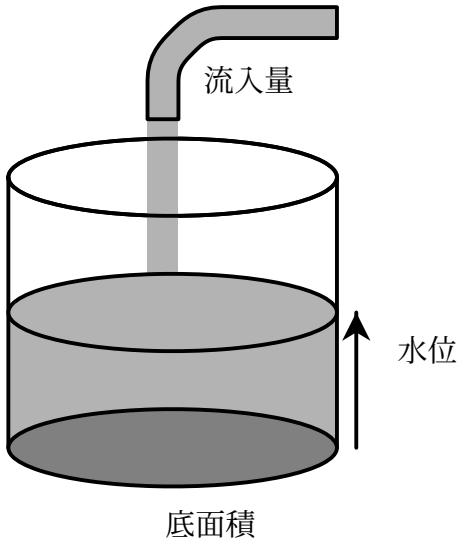


図 1　タンクの問題

1.2 基本１．プログラムの骨格

Java のプログラムはすべて**「クラス (class)」**として記述する必要があります．プログラムの先頭から，中カッコ { } で囲まれた最終行までの

```
public class TankCalculator {
    ...
}
```

がクラスの記述であることを示します[1]．クラス名 TankCalculator 以外の書き方は定型として覚えましょう．

また，java プログラムで何らかの動作をする記述（外部から使うときは命令のように扱えるもの）は**「メソッド」**と呼ばれるものです．その中でも **main** というメソッドは java コマンドで起動された時に呼び出される特別なメソッドです．[2]

class TankCalculator の内側の記述

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

が main メソッドを表します[3]．

`public static void main(String args[])` という記述も定型文として覚えましょう．この内側にあるのがプログラムの本体部分です．

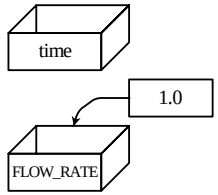
[1] public という語はこのクラスが公開されている（他のクラスからアクセス可能）ことを表します．

[2] public は上記と同じくこのメソッドが公開されていること，static は後述するクラスメソッドであること，void はこのメソッドが返り値を持たないこと，(String args[]) は引数として文字列の配列を取ることを表します．

[3] class TankCalculator の内側の記述であることを分かりやすく示すために，中カッコで囲まれた内側は慣例的に**「字下げ (indent)」**して記述します．プログラムを分かりやすく記述する上での重要なテクニックです．

1.3 基本２．数式と変数

Java の中で数値の計算を行うためには，これを**変数**（数値が入るハコをイメージしてください）に入れて様々な演算を実行します．



【宣言】 変数を使うためには

```
double time;
```

のように「型」と「名前」を記述します．「型」には，整数型，実数型などがあり，用途によって使い分けます（後述）．

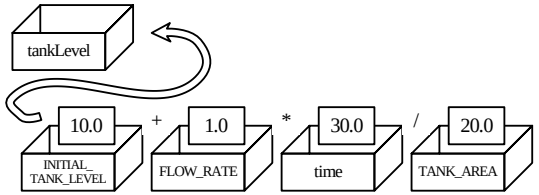


図 2　変数と代入

【定数】また，その値が変わらないものを記号的に記述するには

```
final double FLOW_RATE = 1.0;
```

変数の記述の前に「**final**」を付け，直後に値の代入を書きます [4]．

[4]java では “=” は値の右辺から左辺への代入です．

【計算と代入】式を計算して変数に値を代入するには

```
a = a + 1;
```

```
tankLevel = INITIAL_LEVEL + FLOW_RATE * time / TANK_AREA;
```

のように書きます．加減乗除の四則演算は “+ - * /” で書きます．数学と同じように乗算や除算は加算や減算より優先されます [5]．

[5] 数学では記述を簡略化するために乗算は省略して書けますが，java では省略できません．

【加算】プログラミングでは現在の値に何かを加えるという計算を頻繁に行います．このための演算子　+= が用意されています．例えば TankCalculator.java の 16 行目は現在 30 である time の値を 60 にすることですから，30 を加えると考えれば，

```
time += 30.0;
```

とも書けます．（余裕がある人は確かめてみてください）

1.4 補足： Java における数値の表現

データを表現する基本の型として Java には以下の型が用意されています.

文字	char	１文字を表すための型
論理値	boolean	論理の「真」「偽」を表す型.
整数値	byte, short, **int**, long	整数値を表す型，int は 32 ビット $(-2^{31} \sim 2^{31}-1)$ の long は 64 ビット $(-2^{63} \sim 2^{63}-1)$ の範囲を表せる.
実数値	float, **double**	不動小数点数を表す．もっぱら double を用いる.

同じ数値でも型が違えばプログラムとしては代入などの操作はできなくなる場合があります．例えば double 型の変数の値を int 型の変数には直接代入できません. その場合は以下のような型の変換 (キャスト) が必要になります.

```
int i;
double d = 1.1;
i = (int) d;     /* 型変換により，i には 1 が入る */
```

また，int 型の値の範囲は ±20 億程度です．通常，これで困ることは少ないですが，金額などを扱う場合にはこの範囲を超えることが良くあるので，そのような場合にはより多くの桁を扱える long 型を用います.

1.5 基本３．文字列と端末への出力

Java では文字列定数は "Tank Level" と二重引用符（ダブルコーテーション） " で括って表現します[6]．また，文字列を連結して新しい文字列を作成するには ＋ 演算を用います．例えば

```
"This is " + "a pen"
```

このような式に整数型や実数型の変数を混ぜると，それらの数値を適当なフォーマットで文字列に変換して文字列を作成してくれます．例えば次のように書くと time を文字列に変換して新しい文字列を作ります．

```
"Time = " + time + "s"
```

文字列を端末の画面に出力するには標準ライブラリの “**System.out.println()**” メソッドを使います．() 内に書いた文字列が端末に出力され，最後に改行されます[7]．

[6] java では文字列を扱う変数なども使えます．これには数値と異なり String というクラスを用います．

[7] 改行したくないときは ”System.out.print()” を使います．

1.6 基本４．逐次実行

メソッド内に書かれた代入文などの命令は基本的に上から順に逐次実行されます．コンピュータのプログラムはデータを保持する変数などの作業場所を確保しながらデータに対する操作を 1 ステップづつ逐次実行することに他なりません．

では，もう一度，書いてもらったプログラムを見直してみましょう．

1.7 補足．読みやすいコードを書くための工夫

プログラミングを行っている最中はかなりの意識の集中をしています．しかし，**同じプログラムを時間を置いて読み直すと前に考えていたことが分からなくなる**ことは良くあります．そのために読みやすいプログラムを書くことが重要です．以下はその実践のヒントです．

- 変数の命名：少々長くてもいいですから分かりやすい変数名を付けましょう．変数名を考えることは決して**簡単なことではありません**．文章を推敲するように，その変数名が本当に適切なものかをよく考えましょう[8]．
- 変数名の記述：Java ではプログラムを読みやすくするためのコード化の慣習に従うことが勧められています．例えば次のような慣習です．
  - 変数名，メソッド名は「小文字」で書く．複合語の場合は次の語の先頭を大文字にする．例： `time`, `tankLevel`
  - 定数名は「大文字」で書く．複合語の場合は語と語の間に下線 ”_” を入れる．例：`INITIAL_LEVEL`
  - クラス名は先頭を大文字にする．複合語の場合は次の語の先頭を大文字にする．例：`TankCalculator`

[8] 数学では 1 文字の変数名が好まれて使われますが，プログラミングでは数多くの変数を使うので，短い変数名はそれを使うことが十分理解できる場合にとどめて単語や複合語の変数名を使うべきです．

- 定数の値だけ式に直接書かず，定数を記号的に定義して使う[9].
- java など多くのプログラミング言語では数量の「単位」を明示的に扱えませんが，正しい計算には単位は極めて重要です．コメントなどで単位を明記しましょう.
- 数式の記述において，“=” や “+” などの演算子の前後は読みやすくするために空白を入れましょう．また，演算の優先順位だけに頼らずに分かりにくい所には () を使って優先される演算であることを明示化しましょう.

[9] 直接，プログラムに埋め込まれた数値は「魔法の数字 (magic number)」と呼ばれプログラムの意図を分からなくするとともに，変更を難しくするので嫌われます

1.7.1 補足．変数を使うことのご利益

多くのプログラミング言語では先の例のように数値を変数に代入することによって計算を進めます．ここが電卓と最も異なっているところですし、予め数値の入る「セル」の表が「シート」として用意されている表計算とも異なる所です．変数を使うことのご利益は以下のようなものです.

- 数値を変数に代入することにより同じ数値を異なる計算で何度も利用できる.
- 代入する数値を変えることにより同じ式で異なる場合の計算をできる.
- 変数の名前により計算の意味が分かりやすくなる．[10]

[10] 表計算でセルを A5 などという名前で参照するのとはここが異なります.

1.8 演習（余裕のある人）

上の例は単純な問題ですが，あるシステム（ここではタンク）に流れ込む量（あるいは流れ出す量）は**フロー**，タンクに貯まる水量は**ストック** と呼ばれる量で，システムやそのシミュレーションを考える上で重要な概念です．例えば

1. 年あたり出生数，死亡数と人口
2. 月あたりの収入，支出と預金残高

などはフローとストックとして捉えることができます[11]．TankCalculator.java を参考に 1., や 2. など，フローからストックを求める問題を以下の手順で設定し，プログラミングしてください．手順どおりに行ったことを “□” にチェックして作業を進めてください．

[11] 単位に注目するとストックに対してフローの単位はそれを時間で割ったものになります．

- □ まず現れる変数とその単位を明確化してください．
- □ ストックを計算する式を作成してください．その際，単位の扱いが正当であることを確認してください．
- □ 具体的な計算例を作り，正解を別途，求めておいてください[12]．
- □ これらを java プログラムとして扱う際の変数，定数の型を決め，変数名，定数名を決定してください．
- □ ここまでの作業が済んだら java プログラムを作成し，実行してください．
- □ 計算例について java プログラムが正しく動作することを確認してください．

[12] 予めテストする例を構成しておくことは重要です

ここまでのマトメ

- java プログラミングは，1. エディタでプログラムを書く, 2. `javac` コマンドでコンパイルする, 3. `java` コマンドで実行する，の手順で行う.
- コンパイルや実行に失敗したら，エラーメッセージを参考にデバッグする.
- XXXX.java というプログラムは, 骨格として, `public class XXXX { ... }` を持つ.
- `java` コマンドで実行されるのは, `public static void main(String[] args) { ... }` の中である.
- プログラムは上から下に逐次実行される.
- 命令の終わりにはセミコロン (;) をつける.
- 数値を使うには，`double time;` のように宣言を行う.
- 読みやすさのため，コメント (//) や変数名の付け方には注意を払う.
- 計算と代入を行うには，`a = 2 * b + c;` のようにする.
- 文字列は `"Tank Level"` のようにダブルコーテーションで囲む.
- 文字列の結合，文字列と変数の結合は + で行う.
- 端末の画面に文字列を出力するには以下のようにする.

```
System.out.println("Time = " + time + "s");
```