計算機プログラミング I (1)

プログラムの作成と実行

2000 年 10 月 10 日　増原 英彦

1　履習にあたって

講義情報　講義に関する情報は、http://lecture.ecc.u-tokyo.ac.jp/~masuhara/classes/cpi2000/ からたどれるページに掲示する。このページへは、「教育用計算機システム」のページの「講義用 WWW サーバ」からもたどれる。履習者はブックマークに登録しておくと便利だろう。

履習調整　「計算機プログラミング I」では、1 クラスあたりの履習者を 140 名までに制限している。この授業を履習するためには、次のようにする必要がある。(1) まず ターミナルウィンドウ (kterm) にcp1 5 Returnと入力して、履習の希望を出す。(2) 10 月 23 日以降、10 月 30 日までの間にもう一度、ターミナルウィンドウ (kterm) にcp1 5 Returnと入力して履習の登録をする。(2) の際には、先着 140 名までしか登録が受け付けられないことに注意せよ。この登録方法についての説明は、この授業の web ページからたどれるので、必ず読んでおくこと。

目的と形式　この授業では、プログラミングの基本を知る・計算機科学・ソフトウェア科学の基礎を学ぶことを目的とし、実際に Java 言語を使って学ぶ。履習者は「情報処理」を履習し、計算機の基本的な操作ができることを前提とするが、プログラミングの経験は問わない。

(準) 教科書として以下の本を用い、適宜、補足資料を配布する。(最初の 2 回は教科書がなくても大丈夫である。)

> 岡田謙一・重野寛・古賀祐匠 著,「Java 教科書」, ソフト・リサーチ・センター刊, 1,900 円, 1999 年 (ISBN 4–88373–129–4)

成績はレポート (2 ~ 3 回) と学期末試験を総合して評価する。

2　プログラミングとは何か?

2.1　プログラム

コンピュータはハードウェアとソフトウェアから構成されている。例えば、Playstation 2 や「パソコン」はハードウェアであり、「ドラクエ」や「Windows Me」はソフトウェアである。ソフトウェアは、情報の処理方法をハードウェアに指示するプログラムと、処理の対象となる数値・文章・画像などのデータから成る。ソフトウェアは、それ自体も電子的な情報であるため、必要に応じて作り直したり交換することができる。それゆえコンピュータは様々な目的に利用できる。

ハードウェアが直接理解できるプログラムは、機械語プログラムと呼ばれ、非常に単純な命令を並べたものである。機械語プログラムを人間が正しく書くことは難しい。そのため、こんにちプログラムを作る場合は、人間は (高レベル)[1]プログラミング言語でプログラムを書き、それを機械語プログラムにコンパイル(翻訳) してから実行するのが普通である。このとき、コンパイルの元となるプログラムはソースコード(あるいはソースプログラム) と呼ばれ、コンパイルの結果はオブジェクトコードと呼ばれたりする。

[1]機械語もプログラミング言語の一つではあるが、人間にとって分かりやすいものではないため、低レベルプログラミング言語とも言われる。

2.2 プログラミング言語

(高レベル) プログラミング言語は、自然言語 (人間が日常使う日本語や英語、スワヒリ語などの言語; 人工的に作られたプログラミング言語と区別するためにそう呼ばれる) と同様に、文法規則がある。また、教育向き・研究向き・商用ソフトウェア向き等々、用途によって色々なものがあり、代表的なものだけでも Fortran, C++, Cobol, Pascal, Basic, Perl など数多くある。この授業では、高レベル言語の一つである **Java** 言語を学ぶ。Java 言語は、比較的新しい言語で、数多くの先進的な機能を備えているため、近年急速に普及している。また、システムが無料で公開されているのも特徴である。

多くのプログラミング言語では、ソースプログラムをコンパイルした結果作られるオブジェクトコードは、機械語プログラムである。つまり、CPU (ハードウェアを構成する部品の中でもプログラムを実行する役割を担っているもの) が直接実行できるものが作られるため、無駄なく高速に実行できる。一方で、異なる CPU を搭載したハードウェアでは同じオブジェクトコードが実行できなくなる。例えば、多くの Windows パソコンでは Intel 社製の CPU が使われているが、Playstation では MIPS 社が設計した CPU が使われている。そのため、Playstation 用のゲームは Windows パソコンで動かない。[2]

Java 言語では仮想機械 **(virtual machine)**(あるいは中間コード) と呼ばれる方式によって、CPU などが異なるコンピュータ上で同じプログラムを動かすことができるように設計されている。この方式では、架空のハードウェアを設計しておき、ソースプログラムを、そのハードウェアの機械語へとコンパイルする。そして、Windows パソコンや Unix 環境などのコンピュータでは「そのコンピュータの真似 (simulate) をするソフトウェア」(これを仮想機械と言う) を動かし、その仮想機械がコンパイルされたプログラムを実行する。最近では、電子手帳や携帯電話のコンピュータでも同じ Java プログラムのオブジェクトコードが動くようになっている。

2.3 プログラミングとは?

プログラミングとは、問題が与えられたときに、その問題を解決する手順 (アルゴリズムと呼ぶ) を考え、その手順をプログラムとして表現する (コーディングと呼ぶ) ことである。例えば、「逆行列を求める」という問題に対しては、アルゴリズムとして「ガウス消去法」を始めとする多くの解法がある。コーディングは、ガウス消去法に従って逆行列を計算するプログラムを作ることである。

実際のプログラミングにおいては、アルゴリズムが未知の問題を扱うのが普通であるため、「アルゴリズムを見つける」方法が重要になる。(もちろん、考えたアルゴリズムをプログラムとして表現できなければ意味がないのでコーディングも重要ではある。) そのため、この授業では、単にプログラミング言語の文法や作法を学ぶだけでなく、与えられた問題を解くアルゴリズムを見つける (作る) 方法を学ぶことも目標とする。ただし、書き方が分からないことには、プログラムを作れないので、しばらくは「書き方」が中心になる。

3 プログラムの構造

前に述べたように、(高レベル言語の) プログラムは、人間が読み書きしやすいようになっている。図 1 は、「世界の国からこんにちは。」というメッセージを表示するプログラムである。このプログラムに何が書かれているかを 1 行ずつ見てゆこう。

1. `/* 挨拶するプログラム */` — `/*` から `*/` に囲まれた部分はコメントである。人間が読み書きしやすいように設計されているとはいえ、プログラムを説明抜きで理解することは

[2] 実際には、CPU が違うだけでなく、他のハードウェア部品やオペレーティングシステム (OS) が異なっているためでもある。

```
1  /* 挨拶するプログラム */
2  class Hello {
3    public static void main(String[ ] args) {
4      // ここから実行される
5      System.out.println("世界の国からこんにちは。");
6    }
7  }
```

※ 行番号は説明のためのもので、実際のプログラムには無い。

図 1: 挨拶をするプログラム

難しいので、このようなコメント中に説明を書くのが普通である。(言葉による説明の無い、数式だけの数学の教科書を想像せよ。) この授業においても、プログラムを作る課題を課す予定であるが、その際作成するプログラムには必ずコメントによって説明を加えること。

2. `class Hello {` — 大きなソフトウェアは、小さなプログラムの集まりから作られるのが普通である。Java 言語には、そのような小さなプログラムの単位としてクラスがある。ここでは、「Hello」という名前のクラスを定義することを宣言している。なお、クラスの詳しい説明は学期の後半に行う。

3. `public static void main(String[ ] args) {` — 1 つのプログラム (クラス) はさらに、「ひとかたまりの処理」を行う単位に分割されているのが普通である。Java 言語では、そのような単位としてメソッドがある。ここでは、「main」という名前のメソッドを定義することを宣言している。その前後の `public static void` だの `String[ ] args` は、メソッドの性質を指定しているものである。Java 言語では、「あるクラスを実行せよ」とすると、そのクラスの「main」メソッドを先頭から実行する。メソッドについては学期の中頃に説明する。

4. `// ここから実行される` — これもコメントである。// から行の最後までがコメントになる。

5. `System.out.println("世界の国からこんにちは。");` — これは`System` というクラスが持っている`out` という「もの」に`println` という指示を送れという命令である。

   `("世界の国からこんにちは。")` という部分は、`println` という指示に付随する情報で、パラメータや引数と呼ばれる。

   また、最後の「;」は、文 (命令) の終りを表わしている。これがないとコンパイラは次の行まで文が続いていると思って処理を進めようとしてしまう。

   とりあえずここでは`System.out.println("...");` と書けば、プログラムを実行したターミナルウィンドウ (kterm) 画面にメッセージが表示されると思っておけばよい。

6. `}` — これは 3 行目の開きかっこに対応して、main というメソッドがどこまでかを示している。このように、Java 言語ではプログラム中の範囲を指定するのに中かっこを使う。

7. `}` — これも同様で 2 行目の開きかっこに対応。

結局、このプログラムを何をするものだったのだろうか? 一見複雑そうに見えたが、5 行目だけが実質であり、メッセージを表示するだけのものであった。

4　プログラムの作成と実行

実際に、プログラムを作成・コンパイル・実行する一連の手順を説明する。なお以下では、情報基盤センターの教育用計算機システムの Unix 環境を仮定して説明する。自宅のコンピュータ等で実行する場合は、それぞれの環境に応じた操作に置き換える必要があるかも知れないことに注意せよ。

4.1　ソースプログラムを作る

エディタを使い、`Hello.java` という名前のファイルを作り、図 1 の内容を入力し、保存する。これがソースプログラムである。

⇒ エディタの操作方法を忘れた場合、はいぱーワークブックの「エディタ」を見て復習せよ。また、行の先頭の空白を入力するときは Tab キーを使うよい。

- できれば「計算機プログラム I」のためのディレクトリを作り、そこにファイルを作成するとよい。

  ⇒ ディレクトリの概念・操作を忘れた場合、はいぱーワークブックの「ファイルシステム」を見て復習せよ。

- 大文字・小文字の違いに注意せよ。(ここでは “`Hello`”, “`String`”, “`System`” の先頭だけが大文字である。)

- 漢字コードは EUC コード[3]にして保存する。標準の設定で保存をした場合、漢字コードは EUC になるはずなので、特別な操作は必要ない。

  ⇒ 漢字コードについては、はいぱーワークブックの「日本字入力:漢字コード」の説明を参照せよ。

- ターミナルウィンドウ (kterm) から「`cat Hello.java` Return 」と入力し、入力したとおりの内容が表示されれば ok。

  ⇒ 「`cat`って何?」という人は、はいぱーワークブックの「ファイルシステム: ファイルの内容の表示」を参照せよ。

4.2　コンパイル

- ターミナルウィンドウ (kterm) に「`javac Hello.java` Return 」と入力せよ。コマンド`javac` は Java 言語のコンパイラ (“Java compiler” の略) であり、引数として与えられたソースプログラム (ここでは`Hello.java`) の内容をコンパイルして、Java 仮想機械語のプログラムを作るものである。

- ソースプログラムに問題がなかった場合は、特にメッセージは表示されない。逆に、何かメッセージが表示された場合は、ソースプログラムに誤り (エラー) がある場合である。例えば

```
g987654@ux001> javac Hello.java
Hello.java:5: system は未定義な変数、クラス、またはパッケージ名です。
    system.out.println("世界の国からこんにちは。");
    ^
エラー 1 個
g987654@ux001>
```

[3]Windows や MacOS の環境ではシフト JIS コードにする。

というメッセージの場合、Hello.java というファイルにの 5 行目の付近に誤りがあることを示している。(この例では、「System」の先頭が小文字になっているのが誤り。)

このような場合、ソースプログラムを修正し、再度コンパイルする。

- 問題なくコンパイルできた場合、Hello.class というファイルができているはずである。ターミナルウィンドウ (kterm) から「ls Return 」と入力して確認せよ。このファイルに Java 仮想機械語にコンパイルされたオブジェクトコードが格納されている。Java 言語では、1 つのクラスにこのようなファイルが 1 つ作られるので、クラスファイルと呼ばれる。

4.3 実行

- ターミナルウィンドウ (kterm) に「java Hello Return 」と入力せよ。コマンド java を実行すると、Java 仮想機械が起動される。仮想機械は、引数として指示された名前のプログラム (ここではクラスファイル Hello.class に入っている Java 仮想機械語のプログラム) を読み込み、実行する。
- ターミナルウィンドウ (kterm) に「世界の国からこんにちは。」というメッセージが表示されれば成功である。
- 正しくコンパイルできても、上手く実行できない場合もある。例えば以下のようなメッセージが出た場合、与えられた名前のプログラムを見つけられなかった (hello.class というファイルがなかった) という意味になる。

```
g987654@ux001> java hello
Can't find class hello
g987654@ux001>
```

5 練習

練習 **1-1:** (複数のメッセージ) プログラムの中に、メッセージを表示させる命令 (System.out.println("...");) を追加せよ。どこに追加すると正しくコンパイルできるか? 実行した場合、メッセージはどのように表示されるか?

練習 **1-2:** (println とprint) 上で変更したプログラムの中の、System.out.println と書かれているものをSystem.out.print に変更せよ。(ln がないことに注意!) 実行した場合、メッセージはどのように表示されるか? System.out.println と System.out.print の働きの違いを説明せよ。

練習 **1-3:** (空のprintln) さらにSystem.out.println( ); という行を追加してみよ。実行した場合、メッセージはどのように表示されるか?

練習 **1-4:** (メッセージと計算) System.out.println(2+4); という行とSystem.out.println("2+4"); を追加して、実行せよ。表示の違いから記号「"」(ダブルクオーテーション) の働きを推測し、説明せよ。

練習 **1-5:** (改行・空白) プログラムにとって、改行や空白はどのような意味を持っているのだろうか? 上の練習で作ったプログラムの適当な場所に改行や空白を入れてコンパイル・実行し、どのような変化があるかを見てみよ。特に、(1) 単語の途中 (2) 単語と単語の間 (3) メッセージの途中に改行や空白を入れた場合、取り除いた場合にどうなるか?

6　アンケート (第 1 回 10 月 10 日)

※ このページを切り離し、あてはまるところに丸をつけ、教卓に提出して下さい。

1. 所属している科類は?

   a. 理科 1 類　b. 理科 2 類　c. 理科 3 類　d. 文科 1 類　e. 文科 2 類　f. 文科 3 類

2. 曜日・時限に関らず、この科目 (計算機プログラミング I) を履習しようと思いますか?

   a. 思う　b. まだ分からない　c. 思わない

3. この授業 (火曜日 5 限) の授業を履習しようと思いますか?

   a. 思う　b. まだ分からない　c. 思わない

4. この授業でとり扱ってほしい題材は何ですか? (いくつでも)

   (

)

5. コンピュータの操作にはどの程度習熟していますか?

   a. 難なく使いこなしている　b. 難はあるが、分からないことは自分で調べて使える

   c. 限られたことにしか使えない　d. はっきり言って苦手だ

6. 今日の配布資料についてはどう思いますか? (あてはまるものに全て丸)

   a. 読みやすい　b. 読みにくい (特に　について)

   c. 長い　d. 短い　e. 役に立ちそう

   f. 不要・紙の無駄　g. 字が小さい　h. 紙よりもオンラインで読みたい

   i. その他 (　)

7. 今日の説明についてはどう思いますか? (あてはまるものに全て丸)

   a. 自分よりレベルが高い　b. 自分よりレベルが低い

   c. 聞きやすい　d. 聞きにくい　e. 時間が長い　f. 時間が短い

   g. 分かりやすい　h. 分かりにくい (特に　について)

   i. スライドが見やすい j. スライドが見にくい (改善案:　)

   k. その他 (　)

8. その他、自由に感想を書いて下さい

   (

)

ありがとうございました。可能な限り、次回以降の授業に反映します。