目次



# ★0　オリエンテーション

## ★0.1　この科目について

### 講義概要

Java 言語は，

- 様々な環境で同一のプログラムを動作させることができる
- オブジェクト指向プログラミングの考え方に基づいている
- 利用可能なライブラリが豊富である

といった特徴をもったプログラミング言語です (☆ 1)．この科目では，Java プログラミングの学習を通じて，オブジェクト指向の考え方に触れます．タートルグラフィックスを主な題材とする予定です．また，図形を描画する，マウスでボタンを押したりメニューを選択させる，というような，グラフィックスや GUI（グラフィカルユーザインターフェース）の扱いについても学びます．

☆ 1) JavaScript という，ウェブページの記述などに用いられるスクリプト言語がありますが，これとは全くの別物です．

### 到達目標

Java でプログラムが書けるようになる．オブジェクト指向の考え方を知る．グラフィックスを描画したり GUI を備えたプログラムを作成する方法を学ぶ．

### 系統的履修

**C 言語プログラミングをある程度身につけている**（「計算機基礎実習 I,II」,「プログラミング及び実習」や「応用プログラミング及び実習」を受講している）ことを前提として授業を進めます．

**2017 年度までと教科書が異なります．それにともない，内容にも少し変更があります．再履修者は注意してください．**

### 成績評価の方法

平常点約 20%+定期試験約 80%．平常点は，主に実習課題の達成度にもとづいてつけます（講義中の QUIZ の得点なども含むことがあります）．授業の 3 分の 2 以上に出席していないと定期試験を受けられないことがあります．

### テキストと参考文献

以下を教科書として用います (☆ 2)．必ず入手しておくこと．
立木秀樹, 有賀妙子 「すべての人のための Java プログラミング 第 3 版」（共立出版） 3,000 円 （ISBN:9784320124233）

☆ 2) **2017 年度までは「第 2 版」でした．大幅に改訂されています．**

### いろいろ

- 真剣に授業に参加している人の邪魔をする行為（おしゃべり，途中入退室など）は禁止.
- 大学の授業は，授業時間の他にも自学自習することを前提に作られています．龍谷大学の講義科目の場合，自習時間は講義時間の 2 倍とされています．授業時間以外にも勉強することが必要です.

### 授業計画 (☆ 3)

| | |
|---|---|
| Java 入門その 1 | Java とは |
| Java 入門その 2 | オブジェクトを生成しよう，メソッドを呼び出そう |
| Java 入門その 3 | クラス変数とクラスメソッド |
| Java 入門その 4 | 処理の流れ |
| Java 入門その 5 | 配列 |
| Java 入門その 6 | プリミティブ型と演算子 |
| Java 入門その 7 | クラスを作ろう |
| グラフィックス入門その 1 | GUI クラス (JavaFX) |
| グラフィックス入門その 2 | グラフィックスの描画 |
| グラフィックス入門その 3 | イベント処理 |

☆ 3) 左記の項目ひとつが授業 1 回分，というわけではありません

## ★ 0.2 アクセス

### この科目に関するウェブサイト

高橋のウェブページ (☆ 4)

`https://www-tlab.math.ryukoku.ac.jp/wiki/`

から「時間割」→「科目名」 とたどると，この科目のページにたどりつけます.

☆ 4) 高橋のページにたどりつくには，この URL をブラウザに直接入力するかわりに，理工学部や数理情報学科のウェブサイトからたどったり検索したりする手もありますね.
`www.math.ryukoku.ac.jp`

### 高橋の連絡先など

- 研究室: 1–508（または 1–602）`e-mail: takataka[at]math.ryukoku.ac.jp`
- 高橋の 2018 年度前期の週間スケジュールは以下の通りです．オフィスアワーの曜講時も含めた最新情報はウェブ上または研究室の前に掲出します.

| | 月 | 火 | 水 | 木 | 金 |
|---|---|---|---|---|---|
| 1 | | | | | |
| 2 | Vision | PIP | (学科会議) | 基礎セミナ | |
| 昼休み | | | | | |
| 3 | | | | | |
| 4 | セミナ | | (教授会) | | Graphics |
| 5 | セミナ | | (教授会) | | Graphics |

(ほげ) — あったりなかったりなイベント

ここに示したもの以外に，会議・セミナ・出張等が不定期であります.

# ★1 Java とは

注) この資料中に説明なくページ数が出てきた場合，教科書のページ数を表します．
この資料は，講義中の板書や説明を全部ここに書き込めるほどスペースが広くないかもしれません．「ソースの解説なんかは書き込み式の方がわかりやすいやろ」という思いと「全部書き込み式にするのはいくらなんでも大学の授業としてどうなんや（小学校じゃあるまいし）」という思いが交錯して，中途半端な作りになることがしばしばで….

Java は，次のような特徴をもったプログラミング言語である．

1.

2.

3.

今回の授業では，1. について考える．2. や 3. については，この授業全体を通して学んでいくことになる．

Java の特徴については，知っておくべき大事なことが教科書第 1 章にいろいろ記されている．読んでおくこと (☆ 5)．

☆ 5) この科目の FAQ のページにも情報がある．自分の PC で Java プログラミングできるようにする方法とか．

## ★1.1 とりあえず Java プログラミング

### その前に C 言語プログラミングの例

**G01Hello.c**

```
 1  /* C 言語のプログラムの例 */
 2
 3  #include <stdio.h>
 4
 5  int main(void){
 6    printf("こんちは\n");
 7    printf("a01055 の");
 8    printf("ほげほげおでおま\n");
 9    return 0;
10  }
```

上記のソースファイル G01Hello.c がカレントディレクトリにある場合，計算機室の Linux 環境でこれをコンパイルするには次のようにすればよい．

```
$ cc G01Hello.c
```

このようにコマンド cc を実行すると，G01Hello.c がコンパイルされ，a.out というオブジェクトファイルが作られる．a.out を実行するには次のようにすればよい．

```
$ ./a.out
```

実行結果は次のようになる．

```
こんちは
a01055 のほげほげおでおま
```

### はじめての Java プログラミング

**G01Hello.java**

```
 1  /** はじめての Java プログラム */
 2
 3  public class G01Hello{
 4
 5      public static void main(String[] args){
 6
 7          System.out.println("こんちは");
 8          System.out.print("a01055 の");
 9          System.out.println("ほげほげおでおま");
10
11      }
12
13  }
```

注意: 「クラス」,「メソッド」とは何か，3,5 行目はどういう意味か，等は今後説明します．(☆ 6)

☆ 6) コメントが /** と星 2 つで始まってることには意味がありますが，授業では説明しません．気になる人は教科書を参照してね．

上記の**ソースファイル** G01Hello.java がカレントディレクトリにある場合，計算機室の Linux 環境でこれを**コンパイル**するには次のようにすればよい．

```
$ javac G01Hello.java
```

すると，G01Hello.java がコンパイルされ，**クラスファイル**が作られる．ls コマンドで確認してみよう．

```
$ ls
G01Hello.class  G01Hello.java
```

このクラスファイルを**実行**するには，次のようにすればよい．

```
$ java G01Hello
```

## ★ 1.2 コンパイラ，インタプリタ，仮想マシン

コンピュータのハードウェアや OS(☆ 7) などを合わせた，プログラムが動作する環境のことを，**プラットフォーム**という．ハードウェアの違い (☆ 8)，CPU アーキテクチャの違い，OS の違いなどによって様々なものがある．

☆ 7) OS: オペレーティングシステム

☆ 8) PC，携帯電話，スーパーコンピュータ，etc.

プラットフォームが異なれば実行可能な機械語プログラムの形式も異なるので，機械語プログラムはプラットフォーム毎に用意しなければならない．いわゆる高級言語プログラミングでは，読みにくい機械語プログラムをプラットフォーム毎にいくつも作る，という作業から人間を解放するために，読みやすくかつプラットフォームに（あまり）依存しない**ソースコード**を作成してから機械語プログラムに翻訳する，という手順を踏む．

そのやり方としては，次の二つが代表的である (☆ 9)．

(1) **コンパイラ**を用いる:ソースをまとめて機械語に翻訳する．機械語を直接実行するので，インタプリタを用いるより実行が速い．
(2) **インタプリタ**を用いる: 実行時に逐次的にソースを機械語に翻訳していく．翻訳しながら実行するので遅い．

☆ 9) C 言語ではコンパイラを用いることが一般的．一方，Perl や Ruby のようなスクリプト言語は，インタプリタ型言語である．

開発者が作成したソフトウェアを，多様なプラットフォームを用いる多数のユーザに配布する，という状況を考えると，実行の速さでは (1) に軍配が上がるが，開発や保守の容易さでは (2) が有利である．なぜなら，(1) では開発者がプラットフォームに合わせて個別にコンパイル済みプログラムを用意しなければならないのに対して，(2) ではソースを配布するだけで済むからである (☆ 10)．

☆ 10) ただし，ユーザの方でその言語のインタプリタを自分の環境にインストールしておかねばならないという欠点がある．また，売り物だから等の理由でソースを公開しない，ということが難しいのも欠点となる．

これに対して，Java では，コンパイラとインタプリタの「いいとこどり」をして，次のような手順を採用している．

1. コンパイル時: ソースファイルを，プラットフォームに依存しない中間的な言語（**Java バイトコード**）のプログラムに翻訳する．翻訳してできるファイルを**クラスファイル**という．
2. 実行時: プラットフォーム毎に用意された**仮想マシン** (☆ 11) 上でバイトコードを実行する．

このようにすることで，プラットフォームに依存せずにプログラムを高速に実行できる．

☆ 11) JVM (Java Virtual Machine) と呼ばれる．JVM はインタプリタの一種と考えることもできる．
［発展］近年の Java 仮想マシンでは，プログラムの実行時に動的にバイトコードの一部を機械語にコンパイルする技術（Just In Time Compilation）を採用して性能向上をはかっている．

## ★ 1.3 宿題？

### ★ 1.3.1 QUIZ

今日の授業の内容に関する QUIZ です．

**Q1.** 以下のうち正しいものはどれですか．

1. この授業では教科書を手に入れる必要はない
2. 実習つきの科目なので，その時間内だけコンピュータに向かえば十分である
3. 高橋のオフィスアワーは水 2 と木 3 である
4. わからないことがあるときは，オフィスアワーに高橋を訪ねるとよい

**Q2.** 以下のうち正しいものはどれですか．

1. C 言語のソースファイルは一般にはインタプリタを用いて機械語プログラムに翻訳される．
2. いろんなプラットフォームで Java のプログラムを実行するためには，プラットフォーム毎にソースファイルをコンパイルし直す必要がある．
3. Java 仮想マシンは，プラットフォーム毎に異なる．

**Q3.** `Hoge.java` というソースをコンパイルするにはどうしたらよいですか．

**Q4.** `Hoge.java` をコンパイルするとカレントディレクトリに `Hoge.class` というクラスファイルが作成されたとします．このとき，このクラスファイルを実行するにはどうしたらよいですか．

### ★ 1.3.2 次回までに…

次回は教科書第 2 章の内容を説明します．あらかじめ読んでおくこと．例題のプログラムを作成し実行してみておくこと．