

# 発展プログラミング演習II

## 0. Java言語概要

# 本演習の目的

Java言語の基本を習得する

# なぜJava?

## 需要が多い!!

.....

求人数ランキングを見てみると、最も求人数が多いのは「JAVA」であった。リーマンショック以来、一時期は求人数の落ち込みが見られたが、旧来のWEBアプリケーション開発案件に加え、スマートフォン向けアプリケーション開発でもニーズが高まったことにより、求人数が回復した。

.....

プログラミング言語別 年収・求人数ランキング (2010年版)

【株式会社ワークポート調べ「2010年IT業界の転職市場」】

[http://www.dreamnews.jp/?action\\_press=1&pid=0000027847](http://www.dreamnews.jp/?action_press=1&pid=0000027847)

プログラミング言語別・年収ランキング

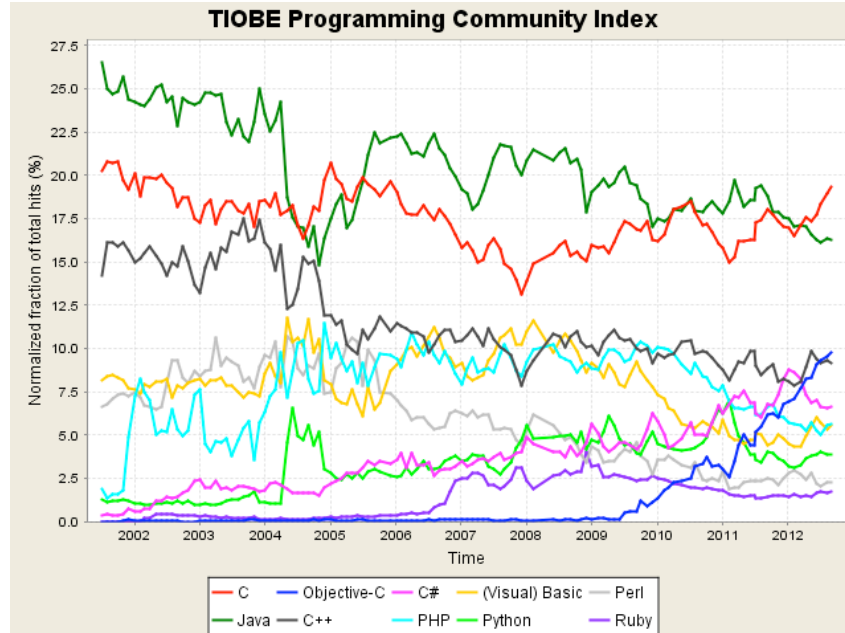
求人数	言語	年収平均値	08年
1位	Objective-C	560.6	圏外
2位	C	543.5	550.3
3位	Perl	533.8	圏外
4位	.net系	531.3	587.0
5位	C++	529.0	556.3
6位	JAVA	526.8	557.0
7位	COBOL	514.3	539.1
8位	C#	509.9	626.9
9位	JavaScript	508.3	538.8
10位	Ruby	504.3	圏外
11位	PHP	502.0	圏外
12位	ActionScript	459.8	圏外

単位・万円

求人数ランキング

求人数	言語
1位	JAVA
2位	PHP
3位	C++
4位	C
5位	Perl
6位	.net系
7位	Ruby
8位	JavaScript
9位	ActionScript
10位	C#
11位	Objective-C
12位	COBOL

# TIOBE Programming Community Index



インターネット上での話題度

発展プログラミング演習II 2012 ©水口・玉田

# Javaの利用範囲

- ・ (通常の) アプリケーションプログラム
- ・ Webアプリケーション
  - ・ アプレット (Webブラウザ上で動作するアプリ)
  - ・ サーブレット (サーバサイド)
- ・ 携帯電話 (iアプリなど)
- ・ デジタル家電
- ・ Android

最先端の分野をカバー

# Javaを学ぶもう一つの理由

## オブジェクト指向プログラミング

詳細は後述

# 授業の方針

- ・ 玉田先生と水口とで2～4回交代でしゃべります
- ・ 座席は自由です
- ・ 念のため学生証はかざしてください
- ・ 適宜課題を出すので指定期日までにmoodleで提出してください
- ・ 時々、授業内提出課題／小テストをやります
- ・ 主に最終テストと最終課題で評価します

# 受講に当たっての注意

- ・ 新しい内容が多いです
  - ・ メモ書き、ノートを取るように
  - ・ 欠席すると穴を埋めるのが大変です
- ・ コピーしてまで課題を提出する意味はまったくありません
- ・ 授業だけでは表面的なことしか学習できません各自でいろいろ試したりして理解を深めること



# 自習の仕方

- ・ 自分でプログラムを書いてみる
- ・ 教科書、Webサイトを見て、手を動かす
  - ・ とりあえず教科書
  - ・ 図書館で借りる
  - ・ 「Java 入門」などで検索する
  - ・ 買う
- ・ 人のプログラムを読んでみる

# 授業補助用Twitter

- ・ @KSUCSEAP2\_2012
- ・ 授業内で補足説明、高度な内容説明、簡単な質問への回答などを行います
- ・ 授業外でも質問等を受け付けます
- ・ 基本的にフォロー返しします
- ・ プライベートと区別したい人は授業用アカウントを作ってください

# 授業計画

Java言語の概要

Java言語の基礎. 変数, 式と演算子.

Java言語の基礎. 条件分岐, 繰り返し.

Java言語の基礎. 配列.

オブジェクト指向言語の基礎. クラスとインスタンス.

オブジェクト指向言語の基礎. フィールド.

オブジェクト指向言語の基礎. メソッド.

オブジェクト指向言語の基礎. 継承.

オブジェクト指向言語の基礎. インターフェース, 例外.

クラスライブラリについて. 標準クラスライブラリの使い方. コレクションとアルゴリズム.

クラスライブラリについて. 標準クラスライブラリの使い方. 入出力.

ソフトウェアとハードウェア. 空きメモリの確認.

応用プログラミング. GUIアプリケーション.

応用プログラミング. ネットワークアプリケーション.

# 例題0.1

1) 次のプログラムを入力せよ（教科書 例2.1改）

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

2) ファイル名をHelloJava.javaとして保存せよ

ディレクトリは各自が課題を管理しやすいように作成するとよい。

3) ターミナルで次のコマンドを実行せよ

```
javac HelloJava.java
```

4) 作業ディレクトリにできているファイルを確認せよ

5) ターミナルで次のコマンドを実行し、出力を確認せよ

```
java HelloJava
```

# MacOSX 10.6 以降での注意

- Javaのデフォルト文字エンコーディングはShiftJISです (10.5以前はUTF-8)
- ShiftJIS以外の文字コードで日本語入りのプログラムを作成するとコンパイルが通りません
- CarbonEmacsの文字コードはUTF-8です (初期状態) MacOSX自体の標準の文字コードがUTF-8です
- 対処法として、以下のコマンドをターミナルで最初に1回実行してください

```
export _JAVA_OPTIONS="-Dfile.encoding=utf8"
```

- 毎回実行するのが面倒くさければ、ホームディレクトリに.bashrcというファイルを作って、そこに上記のコマンドを書いておくと良いでしょう

# チェックリスト

- ・ javacでエラーメッセージが出た場合、必ずプログラムの何かが間違っています
- ・ プログラム中の大文字小文字は合ってるか？
- ・ プログラムのファイル名は  
HelloJava.java  
としたか？
- ・ javacとjavaコマンドの使い方は合ってるか？

# JavaとCのプログラムの比較

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

Java

```
#include <stdio.h>
```

C

```
int main() {  
    printf("Hello, C!\n");  
}
```


- ・基本的な文法は似ている
- ・Javaの方が記述が多い

# JavaとCの実行環境の比較

- javac =コンパイラ
- java ???
- javacでできたファイル HelloJava.class ???
  - ・ 実行ファイルではない!!
- javaコマンドと.classファイルがセットで初めて実行されるようだ



# Javaプログラミングの流れ

- ・ プログラムを書く
  - ・ .javaファイルに保存する。ファイル名は“class”の次の語（**クラス名**）と一致させること
  - ・ コンパイルする `javac <ソースファイル>` ←
  - ・ コンパイルエラーが出たら修正する —
  - ・ 実行する `java <クラス名>`
  - ・ 正しく実行されたか確認する
  - ・ 正しく動作しなかったら修正する —
- 

# 統合開発環境について

- ・ プログラム開発にはエディタ、コンパイラ、デバッガを統合した、統合開発環境(IDE: Integrated Development Environment)が使われることも多い
- ・ プロジェクト管理、バージョン管理、GUI設計ツールなどの支援ツールも含まれている
- ・ Xcode, Eclipseなど
- ・ 本演習では基本を理解するためにテキストエディタ+ターミナルを使う

# 例題0.2

## 1) 次のプログラムを入力せよ

```
import javax.swing.*;

public class HelloJavaSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Hello Java!");
        JLabel label = new JLabel("Hello Java!", JLabel.CENTER);
        frame.getContentPane().add(label);
        frame.setSize(300, 300);
        frame.setVisible(true);
    }
}
```

## 2) プログラムをファイルに保存せよ。ファイル名に注意!!

## 3) コンパイルして実行せよ

※このプログラムは不完全なのでウィンドウのクローズボタンをクリックしても終了しない。

終了させるには実行したターミナルでCtrl+Cを入力するか、メニューバーで終了(⌘Q)を選択する。

# Javaとは

- ・ サン・マイクロシステムズ社（現オラクル社）  
で90年代前半に開発されたプログラミング言語  
および関連技術
- ・ 適用環境に応じて3つのエディション  
(Java SE, Java EE, Java ME)
- ・ 様々なプラットフォームで利用可能  
Solaris, Linux, Windows, MacOSX, 携帯電話, 組み込みシステム, Android ……
- ・ JavaScriptはまったくの別物!!

# Java言語の特徴

- ・ オブジェクト指向プログラミング
- ・ プラットフォーム非依存  
“Write once, run anywhere”
- ・ ネットワーク機能
- ・ “安全な”プログラミング

# オブジェクト指向プログラミング

- ・ オブジェクト = 物 = 物質 + 機能
  - ・ 例：机      天板、脚  
物を上に置く、動かす、倒す、……
- ・ オブジェクト間の相互作用で全体の動作を記述
  - ・ 例：ボールが壁に当たる→ボールが跳ね返る  
オブジェクト指向の場合    どう跳ね返るかはボールと壁が知っている  
手続き型の場合    神様（全体を記述するプログラム）が決める
- ・ オブジェクト = データ + 手続き
  - ・ cf. 手続き型プログラミングでは    データ構造 + アルゴリズム
  - ・ クラス：オブジェクトの定義 = データ構造と、固有の処理を一体化したもの

## 例題0.2

```
import javax.swing.*;

public class HelloJavaSwing {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Hello Java!");
        JLabel label = new JLabel("Hello Java!", JLabel.CENTER);
        frame.getContentPane().add(label);
        frame.setSize(300, 300);
        frame.setVisible(true);
    }
}
```

- ・クラス = オブジェクトの定義
- ・インスタンス = クラスを実体化したもの
- ・フィールド = オブジェクトの内部データ
- ・メソッド = オブジェクトの機能
- ・ただし、クラスHelloJavaSwingは固有のデータ（フィールド）は持っていない  
手続き（メソッド）はmainのみ
- ・JFrame, JLabelの定義はjavax.swing以下でされている

# オブジェクト指向のメリット

- ・ ソフトウェアの再利用がしやすい
- ・ プログラムの保守性がよい
- ・ 大規模なプログラムの作成に適している



# プラットフォーム非依存

- ・ ソースコードは中間言語（バイトコード）にコンパイルされる `.class`ファイル
- ・ バイトコードはJava仮想マシン(JavaVM)で実行される `java`コマンド
- ・ Java仮想マシンはプラットフォームの差を吸収する
  - ・ 但し、現実には微妙に異なる動作をすることがある

# ネットワーク機能

- ・ ネットワークアプリケーション用のライブラリが整備されている
  - ・ TCP/IP インターネットの通信プロトコル
  - ・ XML データの記述・処理
- ・ Web上での実行形態
  - ・ アプレット Webブラウザ上で動作
  - ・ サーブレット Webサーバ上で動作
- ・ 分散オブジェクト

# 安全なプログラミング

- ・ 例外処理

エラー時の処理を明確に記述

- ・ 自動ガベージコレクション

プログラマがメモリを管理する必要がない

- ・ ポインタ操作によるエラーの排除

- ・ メモリ保護機能（サンドボックスモデル）

許可されたメモリやファイル以外にアクセスできない

- ・ 静的な型付け

コンパイル・実行時に厳しく型チェック

# Java言語の弱点

- ・ 実行速度が遅い（かなり改善されている）
- ・ メモリ消費が激しい  
（それほど問題にならなくなってきた）
- ・ プラットフォームのGUIの見かけと異なる  
（設定可能、~~MacOSXの場合アップルが提供~~）

# Java言語でのお約束

- ・ クラス名とソースのファイル名は一致させる。
- ・ 例：クラス名がHelloJava → HelloJava.java
- ・ 基本的には一つのクラスは一つのファイル
- ・ スタンドアロンプログラムを実行させるときに最初に呼ばれるメソッドは必ず

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

(javaコマンドの引数のクラスが持っているmainが最初に呼ばれる)

## 例題0.3

Turtle.javaをmoodleから  
ダウンロードし、同じディ  
レクトリに右のプログラム  
を作成して実行せよ

```
public class Triangle {  
    public void draw() {  
        Turtle t = new Turtle();  
        t.move(10, 200);  
        t.penDown();  
        t.go(100);  
        t.rotate(120);  
        t.go(100);  
        t.rotate(120);  
        t.go(100);  
    }  
  
    public static void main(String[] args) {  
        Triangle tri = new Triangle();  
        tri.draw();  
    }  
}
```

- Turtle.javaはこちらからもダウンロード可能  
<http://www.csg.is.titech.ac.jp/~chiba/lecture/book/>
- 詳しい説明は  
<http://ascii.asciimw.jp/pb/ant/gi4java/sample.pdf>

## 例題0.3のポイント

- ・ タートルグラフィクス：亀を動かして図形を描く
- ・ Turtle: 亀を表すクラス
- ・ Turtle t = new Turtle(); インスタンスの生成
- ・ Turtle.move(int, int) 亀を指定座標に移動させる
- ・ Turtle.penDown() 描画を開始する
- ・ Turtle.rotate(int) 亀の向きを回転させる
- ・ Turtle.go(int) 亀を前進させる

## 例題0.4

例題0.3のプログラムを  
右のように修正して実  
行せよ

```
public class Triangle {  
    public void drawTwo() {  
        Turtle t1 = new Turtle();  
        t1.move(10, 200);  
        t1.penDown();  
        t1.go(100);  
        t1.rotate(120);  
        t1.go(100);  
        t1.rotate(120);  
        t1.go(100);  
  
        Turtle t2 = new Turtle();  
        t2.move(30, 190);  
        t2.penDown();  
        t2.go(70);  
        t2.rotate(120);  
        t2.go(70);  
        t2.rotate(120);  
        t2.go(70);  
    }  
  
    public static void main(String[] args) {  
        Triangle tri = new Triangle();  
        tri.drawTwo();  
    }  
}
```



# 例題0.4のポイント

- ・ 二つのインスタンス t1, t2
- ・ それぞれ別の亀を表している
- ・ このプログラムではt1とt2のメソッドは逐次的に実行される（同時に描画されない）

## 例題0.5

Turtleオブジェクトを使って六角形を描くプログラム  
Hexagon.javaを作成し、コンパイルして実行せよ。

- ・Triangle.javaを改変するとよい。ただし、クラス名に注意。
- ・六角形の大きさはウィンドウに収まる大きさに適当に設定してよい。