Java8 問題集

大森俊太郎 著

理解を深める500問

Javaの概念・プログラミングの
基本・文法・構文がよくわかる！
実力養成に役立つ演習問題を
厳選500問収録！

SCC

Java8 問題集

大森俊太郎 著

理解を深める500問

Javaの概念・プログラミングの
基本・文法・構文がよくわかる！
実力養成に役立つ演習問題を
厳選500問収録！

SCC

Java8 問題集

大森俊太郎 著

理解を深める500問

Javaの概念・プログラミングの
基本・文法・構文がよくわかる！
実力養成に役立つ演習問題を
厳選500問収録！

SCC

・本書に記載されたURL等は執筆時点でのものであり、予告なく変更される場合があります。
・本書の使用（本書のとおりに操作を行う場合を含む）により、万一、直接的・間接的に損害が発生しても、出版社および著者は一切の責任を負いかねますので、あらかじめご了承下さい。

はじめに

Java は、今ではあらゆる分野に、広く適用されています。多くの人が Java を学び、活用しています。

Java に関する情報は、Java の公式サイトから入手することができます。また、その他の Web サイトや書籍からも役立つ情報が得られ、常に最新の環境で学ぶことができます。

しかし、Java を始めたばかりの人、プログラミング経験が少ない人にとって、Java を習得することはなかなか困難です。それには、次のような問題が挙げられています。

(1) Java の概念がよく理解できない。

(2) Java の文法、構文のポイントがつかめず、頭に入らない。

(3) プログラムの基本がよくわからない。

(4) くり返して演習しないと身につかないが、演習問題が少ない。

本書では、このような困難を克服し、ステップアップを図ることができるように、次のような構成になっています。

(1) Java の文法、構文を図式で表現し、ツボを押さえて解説されています。

(2) Java の概念を、比喩を交え、わかり易く解説されています。

(3) Java の仕様についてだけでなく、プログラミングの観点からも解説されています。

(4) Java の概念を把握するための記述式の問題と、それをプログラミングに生かすための演習問題が 500 問用意されていて、知識が身につきます。

なお、本書は Java8 以降に対応しています。したがって、ラムダ式や JavaFX についても学ぶことができます。

本書によって、多くの人が、Java の世界の住人になっていただければ、幸いです。

著　者

【サポートページ】のご案内

下記のサポートページでは、本書掲載の演習問題・実習問題・総合問題の解答をダウンロードすることができます。
また、正誤情報や、補足情報・参考情報などを、必要に応じて掲載します。

本書専用サポートページ
http://www.scc-kk.co.jp/scc-books/support/B-388/support.html

なお、サポートページの内容は、必要に応じて随時更新されますのでご注意下さい。

目　次

第1章

ようこそ Java の世界へ

解　説

Java は、家電製品のソフトウェア開発用として、1995 年にアメリカのサンマイクロシステムズ社によって開発されたプログラム言語およびコンピューティング・プラットフォームです。

当時の新鋭の言語の長所を取り入れ、短所を排除して生まれた Java は、インターネットや科学計算用スーパーコンピュータから携帯電話に至るまで、オブジェクト指向のプログラム言語として、広く使われるようになりました。

現在オラクル社より提供されています。

1-1 Java の特徴

Java には、次の特徴があります。

- オブジェクト指向などの最新のプログラミング技法を取り入れている
- あらゆるＯＳに対応が可能である
- ネットワークに対応している
- マルチスレッドで実行する
- プログラムの安全性が高い

オブジェクト指向などの最新のプログラミング技法

(1) プログラムの処理の流れを、**順次**、**選択**、**繰り返し**という３つの構造で表現する

(2) **オブジェクト指向**：**オブジェクト**という単位で部品化し、オブジェクト同士の相互作用として、システムの振る舞いをプログラミングする

(3) **関数型プログラミング**：処理を関数で記述し、**ラムダ式**の演算として組み合わせてプログラミングする

Java は、これらのプログラミング技法を取り入れた先進的言語で、複雑で大規模なプログラムを効率よく作成できます。

あらゆるＯＳに対応が可能

ハードウェアや OS（オペレーションシステム）が異なると、通常それぞれのＯＳに合わせたプログラミング言語を使い分ける必要があります。

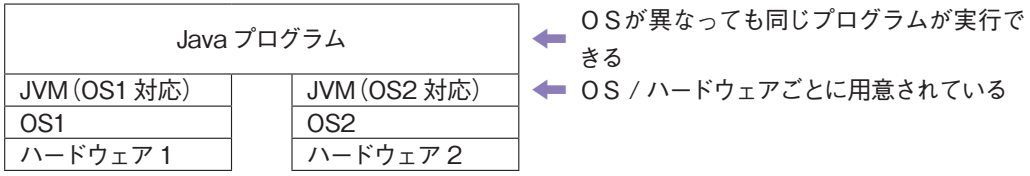
しかし、Java は、仮想マシン（JVM）と呼ばれるプログラムが用意され、ほとんどのＯＳに対応しています。

Java でプログラミングすると、JVM 向けのコード（Java バイトコード）を生成します。

JVM は、Java バイトコードを受け、各ＯＳ対応に動作します。

このためＯＳを意識せず Java プログラムを作成しても、ＯＳの相違を超えて動作することができます。

この原理を ”Write Once, Run Anywhere”（一度書けばどこでも動く）と呼びます。



ネットワーク対応

ネットワーク対応のクラスライブラリが用意されています。

また、Java ベースの強力な UI（ユーザインタフェース）である JavaFX が提供されています。

マルチスレッド

マルチプロセスによる並行処理を行うプログラムを作成することができます。

プログラムの安全性

メモリの割り当てと割り当て解除を Java が管理し、プログラミングのミスによる、メモリ破壊を防止しています。

これをガベージコレクションといいます。

Java

Java というネーミングは、ジャワコーヒーから付けられました。
このテキストも、コーヒーでも飲みながら気楽に進めましょう。

1-2 Java の種類

Java には、3つの分野に向けた3つのプラットフォームが用意されています。

- Java SE（Java Platform Standard Edition）
- Java EE（Java Platform Enterprise Edition）
- Java ME（Java Platform Micro Edition）

それぞれのプラットフォームには、次の機能が含まれています。

Java SE（Java Platform Standard Edition）

標準的な Java プログラミングに適用されます。デスクトップ、サーバー、組込み環境上でも Java アプリケーションを開発できます。

Java はリッチ・ユーザー・インタフェース、パフォーマンス、汎用性、移植性、セキュリティが提供されています。

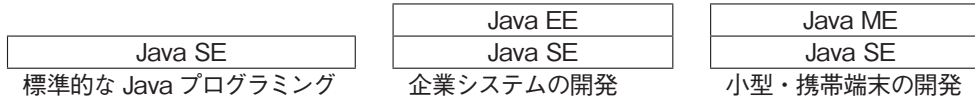
Java EE（Java Platform Enterprise Edition）

プログラミング言語の Java によって企業システムを開発するための標準仕様です。

Java ME（Java Platform Micro Edition）

マイクロコントローラ、センサー、ゲートウェイ、携帯電話、タブレット端末、TV セットトップ・ボックス、プリンタなどのモバイル・デバイスおよび組込みデバイス上で稼働するアプリケーション向けのプラットフォームです。

Java SE, Java EE, Java ME は、次のように組み合わせて使用します。



1-3 躍進する Java

Java は、これまで3度の大きな躍進がありました。

Java の誕生（Java SE）

Java は、これまでにない言語として、注目されて誕生しました。

大きな期待が集まる中、『実行速度が遅くないか』、『適用範囲が限定されるのではないか』など、実用化への道が不安視されていました。

新生 Java、実用化への道（Java 2）

Java は、仕様や仮想マシン（第2章参照）の根本的改革などにより、新しく生まれ変わりました。

不安は期待に変わり、あらゆる分野に信頼できる言語として拡大してきました。

バージョンも次のような呼び名でアップされていきました。

J2SE 1.4, Java SE 5.0, Java SE 6, Java SE 7

プログラミング技法の高度化への道（Java 8）

Java は、更に高度な言語仕様を持つようになり、特にラムダ式や JavaFX に関心と期待が高まっています。本書でも、取り入れています。

ご注意

Java SE, Java 2, Java 8 では、互換性がありません。
それぞれのバージョンで作成された Java プログラムは、他のバージョンでは正しく動かないことがあります。
本書は、Java 8（Java SE 8 以降）に対応しています。

1-4 Java の学習のために

次のように、Java の学習に役立つサイトがいろいろあります。

◆ **Java SE 日本語ドキュメント**

http://www.oracle.com/technetwork/jp/java/index.html

◆ **Java SE API & ドキュメント**

http://www.oracle.com/technetwork/jp/java/javase/documentation/api-jsp-316041-ja.html

◆ **Java(tm) Platform, Standard Edition 8 API 仕様**

https://docs.oracle.com/javase/jp/8/docs/api/

◆ **Java(tm) Platform, Standard Edition 8 ドキュメント**

http://docs.oracle.com/javase/jp/8/

◆ **Oracle Technology Network（日本語）**

http://www.oracle.com/technetwork/jp/index.html

◆ **Java Magazine 日本版**

http://www.oracle.com/technetwork/jp/articles/java/overview/index.html

本書の内容に関連したサイトもあり、操作方法や新しい話題が紹介されています。

また、上記各サイトは適宜変更されますので、検索して適切な情報を入手しましょう。

演習問題

問題１　**次の① - ②に当てはまる言葉を入れましょう。**

Java は、［①］社によって開発されたプログラム言語およびコンピューティング・プラットフォームです。現在［②］社より提供されています。

問題２　**次の① - ②に当てはまる言葉を入れましょう。**

Java は、［①］指向の言語です。さらに、［②］型プログラミングなどの、最新のプログラム技法を取り入れた、先進的な言語です。

問題３　**次の① - ②に当てはまる言葉を入れましょう。**

現在使われている Java は、バージョンから［①］とよばれ、開発キットが［②］償で提供されています。

問題４　**次の① - ②に当てはまる言葉を入れましょう。**

Java は、［①］が異なっても同じプログラムが実行できます。Java は、［②］と呼ばれるプログラムが用意され、ほとんどの［①］に対応しています。

問題５　**次の① - ③に当てはまる語句を (a)-(e) の中から選び、Java の特徴を入れましょう。**

［①］指向
［②］に非依存
［③］の安全性
選択：(a) ネットワーク　(b) プログラム　(c) オブジェクト
　　(d) マルチスレッド　(e) ＯＳ

問題６　**Java はハードウェアやＯＳの相違を超えて動作することができます。この原理を何と呼びますか。**

”Write［①］, Run［②］”（一度書けばどこでも動く）

問題７　**Java をはじめて開発したのは、次の (a)-(e) のうちどこですか。**

選択：(a) Microsoft　(b) Sun Microsystems　(c) IBM　(d) Oracle　(e) Apple

問題８　**現在、Java を提供しているのは、次の (a)-(e) のうちどこですか。**

選択：(a) Microsoft　(b) Sun Microsystems　(c) IBM　(d) Oracle　(e) Apple

問題 9　**次の文は Java の特徴を説明したものです。① - ④に当てはまる語句を (a)-(d) から選びましょう。**

[①]： 並行処理を行うプログラムを作成することができます。
[②]： ガベージコレクションを行います。
[③]： 部品と部品を組み合わせ、再利用してプログラミングします。
[④]： 仮想マシン (JVM) 向けのコード (バイトコード) を生成します。
選択：(a) オブジェクト指向　(b) ＯＳに非依存　(c) マルチスレッド
　　(d) プログラムの安全性

問題 10　**Java の種類を表している略称と説明を (a)-(f) から選択しましょう。**

	略称	説明
① Platform Standard Edition	(　)	[　　]
② Platform Enterprise Edition	(　)	[　　]
③ Platform Micro Edition	(　)	[　　]

選択：(a) Java EE　(b) Java SE　(c) Java ME
　　(d) 標準的な Java プログラム開発
　　(e) スマートフォンやタブレット端末などの開発
　　(f) 企業間で共通のシステムの開発

問題 11　**業界で標準化されているシステムを Java で開発する計画を立てました。Java のプラットフォームとして、何を選択しますか。**

問題 12　**タブレット端末管理システムを Java で開発する計画を立てました。Java のプラットフォームとして、何を選択しますか。**

問題 13　**Java の学習を始めたいと思います。Java のプラットフォームとして、何を選択しますか。**

問題 14　**Java SE は、これまで大きなバージョンアップが行われ、そのバージョンの前後では互換性がなくなりました。転機となったバージョンを① - ③に入れましょう。**

・Java の誕生　[①]
・ 新生 Java、実用化への道　[②]
・ プログラミング技法の高度化への道　[③]

問題 15　**"Java " の名称の由来は何ですか。次の (a)-(e) から選びましょう。**
選択：(a) ミルク　(b) 紅茶　(c) コーヒー　(d) 緑茶　(e) 青汁

第2章

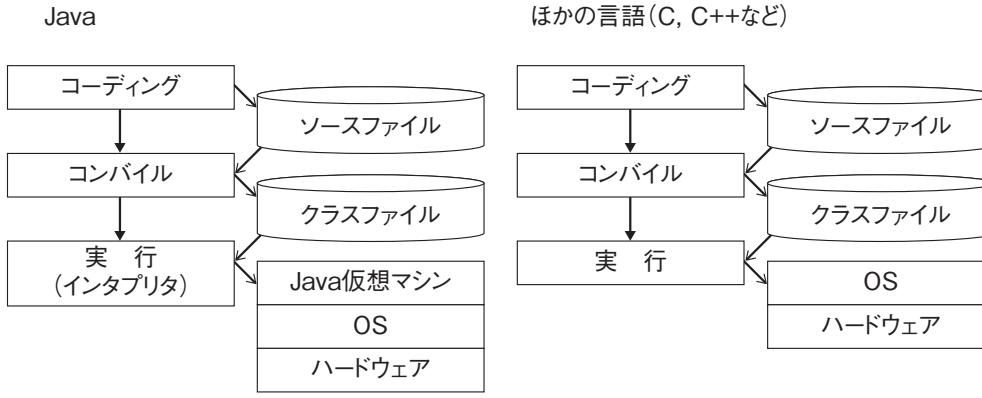
はじめての Java プログラム

2-1 Java の開発／実行環境

プログラミングは、はじめに何をするか、どのようにするかを決める設計を行います。

設計が済むと、次の手順でプログラミングを進めます。

Java と Java 以外の言語（C, C++ など）では、その手順が少し異なります。



開発／実行環境

Java には、次の開発／実行環境が公開されています。

(1) コマンドプロンプト（Windows の基本機能 ）

(2) Java Beans（Oracle 提供 ）

(3) Eclipse（IBM 提供 ）

本書は、コマンドプロンプトでコンパイル、実行することを前提としています。

コーディング

コーディングは、コンピュータに行わせる作業指示 (プログラム) をテキスト形式で作成し、ソースファイルに格納します。

ソースファイルは、人が見て理解するための命令です。

コンパイル

コンパイルは、ソースファイルをコンピュータ（OS）が理解できる形式に変換する作業です。

コンパイルにより得られるファイルの形式は、Java と Java 以外の言語では、異なります。

- **Java 以外の言語**
  コンパイルによってコンピュータ（OS）が直接実行できる実行形式のファイル（**実行ファイル**）を作成します。実行ファイルは、OS によってその形式が変わります。
  Windows の場合は、**ファイル名 .exe** で表されます。

- **Java**
  Java のコンパイルでは、**Java 仮想マシン（JVM）**と呼ばれる、OS に依存しない形式のコードのファイル（**クラスファイル**）が生成されます。
  クラスファイルは、OS が異なっても、（原則として）同じ内容になります。

実行

実行ファイルによる実行では、OS が直接読み込んで実行します。

Java の場合は、OS ごとの専用の Java 仮想マシン（JVM）が、クラスファイルの内容を実行形式に変換しながら OS に渡して実行します。

2-2 コーディング

コーディング

ソースファイルは、開発者自身がテキストエディタで作成します。

プログラムを作成するためにはじめに行うことは、プログラムする人の指令をコンピュータに伝えるための、指令書を作ることです。

この指令書をソースコード（Source code、原始プログラム）と呼びます。単にコードと呼ぶ時もあります。

ソースコードが格納されているファイルをソースファイルと呼びます。

この指令書を作成することを、コーディング（Coding）といいます。

次の文字列のテキストファイルを作りましょう。

ここでファイル名は、1行目のクラス名 "HelloWorld" に拡張子 ".java" を付けて、"HelloWorld.java" とします。

```
public class HelloWorld {

public static void main( String[] args ) {

System.out.println( "hello, world" );

}

}
```

これがソースファイルになります。

プログラマのための記述

コーディングでは、直接クラスファイルのコードに反映される記述以外に、プログラマの便のために行うものもあります。

ここでは、インデントとコメントを取り上げます。

インデント

インデントとは、字下げ (indentation) のことです。

日本語の文章の開始は一字下げて書き始めます。また、章、節、項などは、字を下げて書きます。

このように字下げをすると、内容は変わりませんが、見て内容を理解しやすくなります。

同じように、コードも字下げをすると、人が見たときに内容を理解しやすくなります。

したがって、コーディングでは、インデントが重要です。

次のように、HelloWorld のコードの「{」と「}」の対に注目し、まわりを囲んで見ましょう。

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

このように3重の箱のような構造になっています。

実際のコーディングでは、次のように行の先頭位置をずらせて表現します。

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

このように表現することをインデントといいます。

インデントによってコンパイル結果は変わりませんが、よいプログラミングには必要なことです。

インデントは4字ずつ行うと、見やすくなります。

コメント

プログラマにとってわかりやすいように、ソース上でコードと一緒にコメントを記述することができます。

コンパイラはコメントを無視します。

Java プログラムでのコメントの記述方法は次の通りです。

/*コメント*/

/*と*/ の間にコメントを記述します。複数行にまたがるコメントを記述できます。

// コメント

// の後ろにコメントを記述します。行末までがコメントとなります。
単一行やコードの後ろにコメントを記述できます。

/コメント*/**

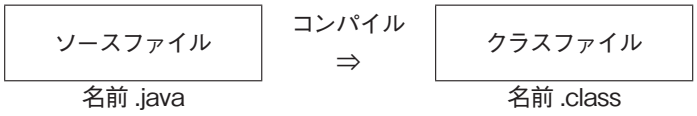
/**と*/ の間にコメントを記述します。
javadoc ツールを使うと、/**と*/ の間のコメントを取り出して、文書（ドキュメント）を作成することができます。

2-3 コンパイル

コンパイル

Java コンパイラはソースファイルをクラスファイルにコンパイル（翻訳）します。

クラスファイルには、Java 仮想マシンに対するバイトコードが生成されます。



名前 .java　　名前 .class

操作は次のように行います。

Windows の場合、次のように操作します。

(1) ソースファイルが格納されているフォルダのパスを、エクスプローラでコピーします。

(2) コマンドプロンプトを起動します。

(3) 次のコマンドを入力します。

> cd パス　　（パスには、コピーしたパス情報を貼り付けます）

パス > `javac HelloWorld.java`

HelloWorld.class ファイルが作成されます。

ここで、javac などのコマンドが存在するフォルダ（ディレクトリ）のパス（PATH）が、OS に設定されている必要があります。

Java のコンパイル結果であるクラスファイルは、他のＯＳ／ハードウェアの Java 仮想マシンでも有効です。

2-4 実行

Java 仮想マシン

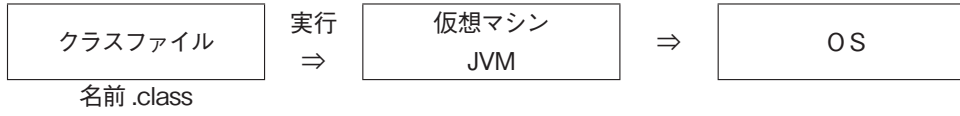
Java では仮想的なマシンを想定します。この仮想的なマシンを Java 仮想マシン(JVM)と呼びます。

Java コンパイラは、JVM に対するコードを生成します。

ターゲットマシンに対しては、JVM のコードを変換しながら実行します。

インタプリタ

バイトコードを、実行可能な形式に逐次変換しながらマシンBを実行するものを、インタプリタといいます。「Java 仮想マシン（JVM)」という言葉は、インタプリタの意味で使われることもあります。



名前 .class

コマンドプロンプトで次のように操作します。

パス > `java HelloWorld`

クラスファイルが存在するフォルダで実行します。

実行時に「エラー : メイン・クラス HelloWorld が見つからなかったかロードできませんでした」というエラーが発生することがあります。この場合、ソースに「package パッケージ名 ;」というコードが指定されているかを調べ、「// package パッケージ名 ;」とすることでエラーを回避できます。(package については「第 12 章 クラスの関係を深める」で解説します)

この HelloWorld はクラス名であり、ファイル名ではないので、拡張子 "class" は指定しません。

ここで、java などのコマンドが存在するフォルダ (ディレクトリ) のパス (PATH)、およびクラスファイルのパス（CLASSPATH）が、OS に設定されている必要があります。

各 OS の Java 仮想マシン（インタプリタ）が、オラクル社および各社から無償で提供されています。

2-5 完成に向けて（テスト／デバッグ）

プログラムは、コーディング完了後、次の手順で完成を目指します。

(1) コーディングされたソースを見直し、エラーを見つける。

(2) コンパイル時にエラーが表示された場合、エラー内容を検討してソースを修正する。

(3) 実行して結果を確認し、不具合があれば修正する。

結果の確認のため、次の条件を含むデータで実行するようにします。

・正常なケース
・境界
　データに制限がある場合、最大の値あるいは最小の値のデータ
・異常ケース
　最大を超える値のデータ等

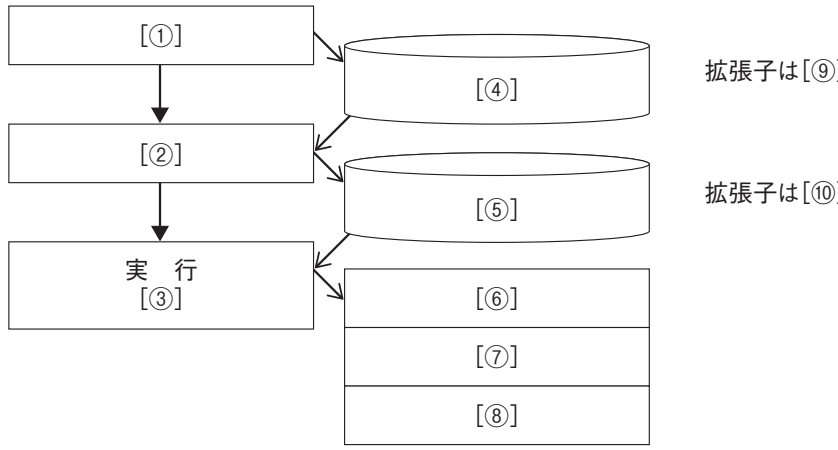
プログラムのエラー、不具合をバグ（虫）、バグを見つけて修正することをデバッグ（虫取り）といいます。

バグ、デバッグ

むかし、むかし、あるところに動きがおかしくなったコンピュータ（ハードウェア）がありました。調べてみると中に虫（bug）が入り込んでいました。その虫を取り除く（debug）と、コンピュータは正常に動きました。それから、プログラミングの世界でも、バグ、デバッグという言葉が使われるようになりました。

演習問題

問題 16　**①-⑩に当てはまる語句を (a)-(k) の中から選びましょう。**



選択：(a) インタプリタ　(b) java　(c) class　(d) コーディング
(e) クラスファイル　(f) ハードウェア　(g) コンパイル　(h) Java 仮想マシン
(i) ソースファイル　(j) 実行ファイル　(k) ＯＳ

問題 17　**次の①-③に当てはまる言葉を入れましょう。**

Java 言語の［①］は、仮想的なコード［②］を生成します。実行は［③］が［②］を逐次解釈します。他の言語 (C 言語など) の［①］は、ターゲットマシンのコードを直接生成し、実行します。

問題 18　**次の①-③に当てはまる言葉を入れましょう。**

Java 言語のコードを作成することを［①］といいます。［①］は実行に直接関係のあるものと、プログラマの便のためのものがあります。後者には、［②］と［③］があります。

問題 19　**次の［ ］に記号を入れ、コメントを完成しましょう。①-④に当てはまる言葉を入れましょう。**

① ［ ］
　　数行にわたるコメント
　［ ］
② ［ ］1行のコメント
③ ［ ］1行のコメント ［ ］
④ a = b;　［ ］コードのうしろにコメント

問題 20　**次の① - ③のコメントの使い方を説明している文を、(a)-(c) から選びましょう。**

① a = b; // コメント
② // a=b;
③ /**
　　文書記述
　*/

選択：(a) Javadoc ツールでドキュメントを生成するための記述
　　(b) 削除したコードがわかるようにコードをコメントにした記述
　　(c) コードがどのような働きをするかの説明

問題 21　**次のファイルの拡張子を付けましょう。**

ソースファイル . [①]
クラスファイル . [②]

問題 22　**ソースファイルのコンパイルから実行までの操作を完成しましょう。**

> [①] HelloWorld.java
> [②] [　　③　　]
Hello, world

問題 23　**次のソースプログラムを、インデントして見やすくしましょう。**

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

問題 24　**HelloWorld.java プログラムのコンパイルが正常に終了したときに、このプログラムを実行するために使うコマンドはどれですか。**

① HelloWorld
② javac HelloWorld.java
③ java HelloWorld.java
④ javac HelloWorld
⑤ java HelloWorld

問題 25　**Java で有効なコメント用区切り記号はどれですか。**

① //
② /*　*/
③ !--
④ <!--　-->
⑤ /**　*/

問題 26　**Java コンパイラが、直接対応するコンピュータの機械語を生成しないのはなぜですか。**

問題 27　**次のソースをコンパイル、実行したときに、何が表示されますか。**

```
public class Comment01 {

    public static void main( String[] args ) {

        System.out.println( "print#1" );

     // System.out.println( "print#2" );

        System.out.println( "print#3" );

     /* System.out.println( "print#4" );

     // System.out.println( "print#5" );

        System.out.println( "print#6" );

     /* System.out.println( "print#7" ); */

        System.out.println( "print#8" );

    }

}
```

問題 28　**次の① - ③に当てはまる言葉を入れましょう。**

次のコンパイル操作のうち、正しいものはどれですか。

① ソースプログラムを「Sample.txt」で作成し、「javac Sample.txt」とコンパイルした。
② ソースプログラムを「Sample.txt」で作成し、ファイル名を「Sample.java」に変えて「javac Sample.java」とコンパイルした。
③ Mac で作成したソースファイル「Sample.java」を、Windows でコンパイルした。

問題 29　**次のコンパイル／実行操作のうち、正しいものはどれですか。**

② Mac でコンパイルしたクラスファイル「Sample.Class」を、Windows で実行した。
② ソースファイル「Sample.java」を、「Sample.Class」にファイル名を変えて実行した。

問題 30　**①-⑧の語句と関係の深いものを (a)-(h) から選びましょう。**

① Java 仮想マシン　② Java コンパイラ　③ インデント　④ クラスファイル
⑤ ソースファイル　⑥ インタプリタ　⑦ コーディング　⑧ ドキュメント化
選択：(a) javac　(b) 逐次解釈　(c) /**コメント*/　(d) 字ずらし
(e) java　(f) JVM　(g) class　(h) テキストエディタ

実習問題

実習問題は、実際にパソコンを操作して進めましょう。

問題 31　**解説編のソースファイル HelloWorld.java をコーディング、コンパイル、実行して次の文字列が表示されることを確認しましょう。**

```
hello, world
```

問題 32　**次のソースファイルをコンパイルしたところ、コンパイルエラーとなりました。**

ソースファイル名を変えずに正しくコンパイルしましょう。

表示結果：`hello, world`

ソースファイル名：`HelloJava.java`

```
public class HelloWorld {

    public static void main( String[] args ) {

        System.out.println( "hello, world" );

    }
}
```

問題 33　**HelloJava.java の表示結果を次のように変更しましょう。**

```
hello, java
```

問題 34　**HelloJava.java の表示結果を次のように変更しましょう。**

```
世界のみなさん　こんにちは
ようこそ　Java の世界へ！
```

問題 35　**次のソースファイルをコンパイルしたところ、コンパイルエラーとなりました。**

デバッグして実行しましょう。

ソースファイル名：Goukaku.java

```
public class goukaku {
    public static void main( string[] args ) {
        System.out.println( 合格 )
    }
}
```