

第十二回 演習問題（オブジェクト指向）

諸注意

- 「Human.java」, 「Japanese.java」, 「American.java」, 「Englishable.java」, 「Fukuijin.java」の5ファイルをWebから提出する。
- 指定した処理が正常に動くようであれば、適宜独自メソッド等を実装してくれて構わない。
- コピペ発覚時は見せた側も見せてもらった側も両方0点とする。
- 必ず**コンパイルエラーのない状態で提出すること**（自動採点したいのでコンパイルエラーがあると、全て0点になってしまう）。
- 課題の途中で提出することになった場合、**コンパイルエラーさえ出なければ、課題の途中の状態で提出してくれて構わない**。一部のメソッドだけが実現できていない場合、コンパイルエラー出ないならばそのままの状態で提出してくれてよい。
- 主にコンソール出力で評価しているため、デバッグに用いたようなコンソール出力が残っていないように気をつけること。**基本的にコンソール出力を指定しない限りは、課題内でコンソール出力はないものとする**。
- **Package は使わないこと**（デフォルトパッケージで実装する）。Package で実装すると、自動採点がうまくいきません。

課題 1

1	問題	WebClass（2限目の方）に「プロ IV_第十二回_演習問題」の題でオブジェクト指向に関する練習問題を準備した。これに解答せよ。なお、練習問題の得失点は演習の成績には影響しないため、資料を見ずに自身の能力を測ってみてほしい。練習問題に1回以上解答していれば実施したとみなし、本演習の30点分とカウントする。
---	----	---

課題 2

2-1	問題設定	日本人やアメリカ人を管理するシステムを開発する。上司から下図のクラス設計が与えられた。コレを実装せよ。
	クラス図	<pre> classDiagram class Human { String name Human mother Human father +abstract void greeting() +Human getMother() +Human getFather() } class Japanese { void greeting() } class American { void greeting() } Human < -- Japanese Human < -- American </pre> <p>図 1. クラス図</p>
	諸注意	<ul style="list-style-type: none"> ● フィールドは全て隠蔽すること（ただしサブクラスからのアクセスは許容する）。 ● コンストラクタは各フィールドを初期化する。定義は親クラス側に記述する。 ● 子クラス側にもコンストラクタは記述するが親クラスのそれを流用するものとする（二重定義は避ける）。 ● greeting() はクラスによって異なるので以下の書式のコンソール出力を実装する。
	greeting() の出力	<p>Japanese(name="田中"の場合) こんにちは 田中はお辞儀をした</p> <p>American(name="Mike"の場合) Hello Mikeはハグをした</p>

2-1	テスト例	Main.javaのmain()に書く場合 Japanese takeshi = new Japanese("やまとたけし", null, null); Japanese nobue = new Japanese("やまものぶえ", null, null); Japanese takashi = new Japanese("やまとたかし", nobue, takeshi); American john = new American("John", null, null); American cindy = new American("Cindy", null, null); takeshi.greeting(); nobue.greeting(); takashi.greeting(); john.greeting(); cindy.greeting(); takashi.getFather().greeting();
	テスト例 出力	こんにちは やまとたけしはお辞儀をした こんにちは やまものぶえはお辞儀をした こんにちは やまとたかしはお辞儀をした Hello Johnはハグをした Hello Cindyはハグをした こんにちは やまとたけしはお辞儀をした
	採点基準	1. ソースコードが諸注意を満たしている。 2. Japaneseをインスタンス化でき、greeting()が適切に動作する。 3. Americanをインスタンス化でき、greeting()が適切に動作する。 4. getMother(), getFather()が適切に動作する。

2-2	問題設定	課題 2-1 ではそれぞれに対して一々greeting()を実行せねばならず面倒であった。全員挨拶してくださいといえは挨拶してくれるので、ArrayList で一元管理したい。Human クラスを以下の要件に従って拡張せよ。
	要件	<p>以下、全て static で実装せよ。</p> <ul style="list-style-type: none"> 人間を管理する ArrayList 型フィールドを定義せよ。 リストに Human を登録する void add(Human h) メソッドを実装せよ。 登録されている人間全員に挨拶させる void allGreeting() メソッドを実装せよ。 人名からインスタンスを取得するメソッド Human getByName(String name) を実装せよ。本メソッドは既にリストに登録されている人の名前を指定することで、そのインスタンスを返すものである。登録されていない場合は null を返すこと。
	諸注意	<ul style="list-style-type: none"> 人名は同姓同名はないものとする。 ArrayList のジェネリクス型は明示していないが適切なものを記述すること（何も書かないは×）。
	テスト例	<p>Main.javaのmain()に書く場合</p> <pre>Human.add(new Japanese("やまもとたけし", null, null)); Human.add(new Japanese("やまとのぶえ", null, null)); Human mother = Human.getByName("やまとのぶえ"); Human father = Human.getByName("やまもとたけし"); Human.add(new Japanese("やまもとたかし", mother, father)); Human.add(new American("John", null, null)); Human.add(new American("Cindy", null, null)); Human.allGreeting(); Human takashi = Human.getByName("やまもとたかし"); takashi.getMother().greeting();</pre>
	テスト出力例	<pre>こんにちは やまもとたけしはお辞儀をした こんにちは やまとのぶえはお辞儀をした こんにちは やまもとたかしはお辞儀をした Hello Johnはハグをした Hello Cindyはハグをした こんにちは やまとのぶえはお辞儀をした</pre>
	採点基準	<ol style="list-style-type: none"> add(), allGreeting()の動作が要件どおりである。 getByName()の動作が要件どおりである。

2-3	問題設定	課題 2-2 の拡張として、Fukuijin クラスと Englishable インタフェースを実装する。福井人とアメリカ人だけ英語が話せる (Englishable) とする (なぜだかは言及しない)。
	<pre> classDiagram class Human { <<abstract>> String name Human mother Human father abstract void greeting() Human getMother() Human getFather() } class Japanese { void greeting() } class American { void greeting() void greetingEng() } class Fukuijin { void greeting() void greetingEng() } class Englishable { <<interface>> void greetingEng() } Human < -- Japanese Human < -- American Japanese < -- Fukuijin Fukuijin < -- Englishable American < -- Englishable </pre> <p>図 2. クラス図</p>	
	諸注意	<ul style="list-style-type: none"> ● 諸注意は 2-1 と同じとする。 ● greeting() と greetingEng() はクラスによって異なるので以下の書式のコンソール出力を実装する。
	greeting() の出力	Fukui(name="福井"の場合) ええてんきでえ～ 福井はお辞儀をした
	greetingEng() の出力	Fukui(name="福井"の場合) はろお～ 福井はお辞儀をした American(name="Mike"の場合) (greeting()の結果を出力する。)
	テスト例	Main.javaのmain()に書く場合 Fukuijin takashi = new Fukuijin("やまとたかし", null, null); Human.add(takashi); Human.add(new Fukuijin("やまとたかお", null, takashi)); Human.add(new American("Cindy", null, null)); Human.allGreeting(); takashi.greetingEng();
	テスト出力例	ええてんきでえ～ やまとたかしはお辞儀をした ええてんきでえ～ やまとたかおはお辞儀をした Hello Cindyはハグをした はろお～ やまとたかしはお辞儀をした
	補足	福井弁を貶める意図ではないことをご理解いただきたい。 (「こんにちは」の訛りをググったら知恵袋で「ええてんきでえ～」が出てきたのである。)
	採点基準	<ol style="list-style-type: none"> 1. コード内に必要なキーワードが記述されている。 2. Fukuijinのインスタンス化とgreeting()の動作が要件どおりである。 3. FukuijinとAmericanとgreetingEng()の動作が要件どおりである。

宿題

次週以降は自由開発課題の時間となる。来週までに開発するシステムを具体的に考えてくることを宿題とする。また、それを文書化し、適切なクラス設計を考えてくるとなると良い。もちろん先に開発を進めてくれていても良い。

7月10日、17日の授業は以下の予定である。

- 2 限目：開始直後（10 時 35 分頃）に出席確認を兼ねて「開発するシステム概要」を提出してもらう。それ以降は自由開発時間として、質問を受け付ける。11 時半～は昼食にしてくれても良い。
- 3 限目：開始直後（13 時 5 分頃）に出席確認を兼ねて「現在の進捗」を提出してもらう。それ以降は自由開発時間として、質問を受け付ける。14 時～は解散してくれても良い。

最終的な提出物（動画付き PPT ファイル）の締切は7月20日（金）の24時とする。これ以降は結合作業等々があるため受け付けられない。この提出がなかった人は当然ながら自由課題が 0 点となり落単は避けられないだろう。