第十二回　演習問題（オブジェクト指向）

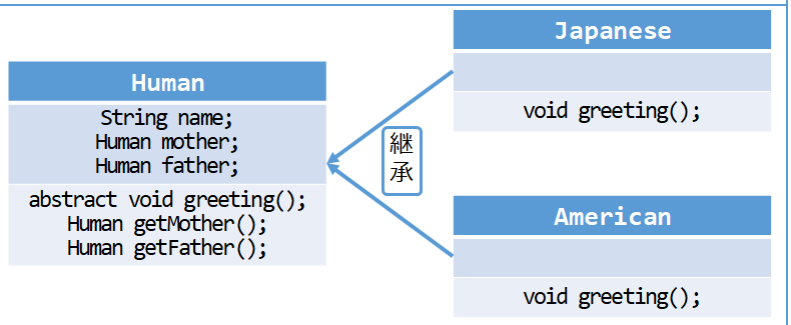
諸注意

- 「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	 図１．クラス図
	諸注意	● 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	**Japanese(name=”田中”の場合)** こんにちは 田中はお辞儀をした **American(name=”Mike”の場合)** Hello Mikeはハグをした

<table>
<tr><td rowspan="3">2-1</td><td>テスト例</td><td>Main.javaのmain()に書く場合
<pre>
Japanese takeshi = new Japanese("やまもとたけし", null, null);
Japanese nobue = new Japanese("やまもとのぶえ", null, null);
Japanese takashi = new Japanese("やまもとたかし", 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();
</pre></td></tr>
<tr><td>テスト例
出力</td><td>こんにちは
やまもとたけしはお辞儀をした
こんにちは
やまも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
やまもとたけしはお辞儀をした</td></tr>
<tr><td>採点基準</td><td>1．ソースコードが諸注意を満たせている．
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．</td></tr>
</table>

課題 2-1 ：解答例（Human.java）

```
public abstract class Human {
	// 継承前提なことを考えるとprotectedにしておくほうが良い
	protected String name;
	protected Human mothre;
	protected Human father;

	// 初期化するだけのコンストラクタ
	public Human(String name, Human m, Human f){
		this.name = name;
		this.mothre = m;
		this.father = f;
	}

	// getter
	public Human getMother(){ return this.mothre; }
	public Human getFather(){ return this.father; }

	// greetings()は継承先で定義される抽象メソッド
	public abstract void greeting();
}
```

課題 2-1 ：解答例（Japanese.java）

```
public class Japanese extends Human{
	// コンストラクタは親クラスのものを利用する
	public Japanese(String name, Human m, Human f) {
		super(name, m, f);
	}

	// 適切にgreeting()をオーバーライドする.
	@Override
	public void greeting() {
		System.out.println("こんにちは");
		System.out.println(this.name + "はお辞儀をした");
	}
}
```

課題 2-1 ：解答例（American.java）

```
public class American extends Human{
	public American(String name, Human m, Human f) {
		super(name, m, f);
	}

	@Override
	public void greeting() {
		System.out.println("Hello");
		System.out.println(this.name + "はハグをした");
	}
}
```

<table>
<tr><td rowspan="5">2-2</td><td>問題設定</td><td>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ば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．</td></tr>
<tr><td>要件</td><td>以下，全て static で実装せよ．
● 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．登録されていなかった場合は null を返すこと．</td></tr>
<tr><td>諸注意</td><td>● 人名は同姓同名はないものと考える．
● 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×）．</td></tr>
<tr><td>テスト例</td><td>Main.javaのmain()に書く場合
Human.add(new Japanese("やまもとたけし", null, null));
Human.add(new Japanese("やまもとのぶえ", null, null));
Human mother = Human.getByName("やまもとのぶえ");
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();</td></tr>
<tr><td>テスト
出力例</td><td>こんにちは
やまもとたけしはお辞儀をした
こんにちは
やまも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
やまもとのぶえはお辞儀をした</td></tr>
<tr><td>採点基準</td><td>1. add(), allGreeting()の動作が要件どおりである．
2. getByName()の動作が要件どおりである．</td></tr>
</table>

課題 2-2：解答例（Human.java に以下を追記する）

```
// Humanの一覧を管理するためのリスト
private static ArrayList<Human> list = new ArrayList<Human>();

// リストにインスタンスを追加する.
public static void add(Human h){
	list.add(h);
}

// リスト内全員にgreeting()を実行させる.
public static void allGreeting(){
	for(Human h : list){
		h.greeting();
	}
}

// リスト内から名前が一致するHumanインスタンスを取得する.
public static Human getByName(String name){
	for(Human h : list){
		if(h.name.equals(name)){
			return h;
		}
	}
	return null;
}
```

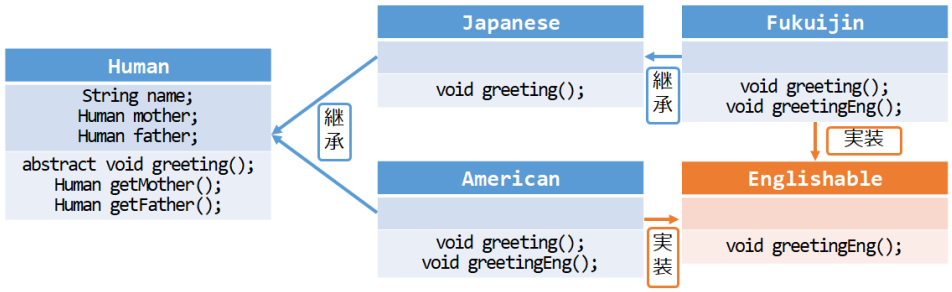
<table>
<tr><td rowspan="9">2-3</td><td>問題設定</td><td>課題 2-2 の拡張として，Fukuijin クラスと Englishable インタフェースを実装する．福井人とアメリカ人だけ英語が話せる（Englishable）とする（なぜだかは言及しない）．</td></tr>
<tr><td colspan="2"> 

図 2．クラス図</td></tr>
<tr><td>諸注意</td><td>● 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．</td></tr>
<tr><td>greeting()
の出力</td><td>Fukui(name=”福井”の場合)
ええてんきでぇ～
福井はお辞儀をした</td></tr>
<tr><td>greeting
Eng()
の出力</td><td>Fukui(name=”福井”の場合)
はろぉ～
福井はお辞儀をした

American(name=”Mike”の場合)
（greeting()の結果を出力する．）</td></tr>
<tr><td>テスト例</td><td>Main.javaのmain()に書く場合
Fukuijin takashi = new Fukuijin("やまもとたかし", null, null);
Human.add(takashi);
Human.add(new Fukuijin("やまもとたかお", null, takashi));
Human.add(new American("Cindy", null, null));
Human.allGreeting();
takashi.greetingEng();</td></tr>
<tr><td>テスト
出力例</td><td>ええてんきでぇ～
やまもとたかしはお辞儀をした
ええてんきでぇ～
やまもとたかおはお辞儀をした
Hello
Cindyはハグをした
はろぉ～
やまもとたかしはお辞儀をした</td></tr>
<tr><td>補足</td><td>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「ええてんきでぇ～」が出てきたのである．）</td></tr>
<tr><td>採点基準</td><td>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．</td></tr>
</table>

課題 2-3：解答例（Englishable.java）

```
public interface Englishable {
	// interfaceなのでpublicと書かなくてもpublicになる
	void greetingEng();
}
```

課題 2-3：解答例（Fukuijin.java）

```
public class Fukuijin extends Japanese implements Englishable{
	public Fukuijin(String name, Human m, Human f) {
		super(name, m, f);
	}

	@Override
	public void greeting() {
		System.out.println("ええてんきでぇ～");
		System.out.println(this.name + "はお辞儀をした");
	}

	@Override
	public void greetingEng() {
		System.out.println("はろぉ～");
		System.out.println(this.name + "はお辞儀をした");
	}
}
```

課題 2-3：解答例（American.java）

```
public class American extends Human implements Englishable{
	public American(String name, Human m, Human f) {
		super(name, m, f);
	}

	@Override
	public void greeting() {
		System.out.println("Hello");
		System.out.println(this.name + "はハグをした");
	}

	@Override
	public void greetingEng() {
		this.greeting();
	}
}
```

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お良い．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．