プログラミング応用演習ｂ　第３回　演習課題ヒント

プログラム仕様書作成課題

課題クラスを読み、次に示すクラスの仕様書を完成させよ。なお、仕様書は、クラス１つに付き１つ作成する。加えて、図1のようなクラス継承の模式図を作成せよ。

＜【クラス名】 のプログラム仕様書＞

作成者：【学籍番号】【名前】

（1）クラス
クラス名：【クラス名】
説明：【クラスが何を表現しているか、どのように利用されるか】
スーパークラス：【継承したクラス名、スーパークラスがない場合は「なし」】

（2）フィールド
変数名：【変数名】
・属性：【アクセスレベル】、【変数の種類（static 変数、非 static 変数、ローカル変数）】、【型】
・フィールドの目的：【何の為に用意されたフィールドなのか、どのように利用されているか】

（3）コンストラクタ
コンストラクタ名：【コンストラクタ名】
・引数：【引数名】、【型】、【変数の目的】
・処理内容：【処理の内容を記述、処理が複数の手順に分けられる場合、処理する順に箇条書きで列挙する】

（4）メソッド
メソッド名：【メソッド名】
・属性：【アクセスレベル】、【変数の種類（static 変数、非 static 変数、ローカル変数）】、【型】
・戻り値の型：【型】
・戻り値の内容：【処理した結果、何を return すればよいか】
・引数：【引数名】、【型】、【変数の目的】
・処理内容：【処理の内容を記述、処理が複数の手順に分けられる場合、処理する順に箇条書きで列挙する】
・オーバーライドの有無：【有 or 無、スーパークラスのメソッドをオーバーライドしたか】

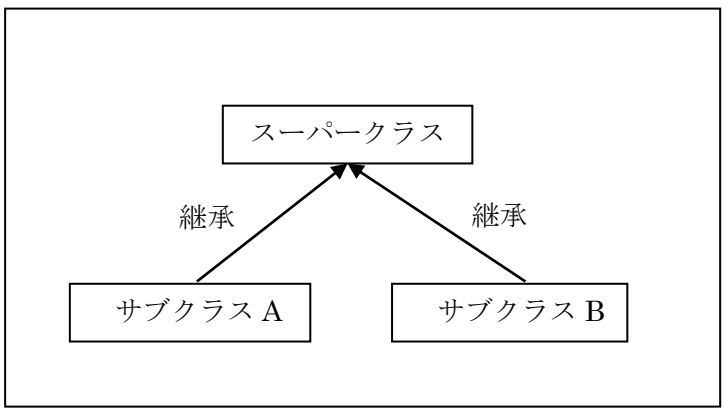

図 1. クラス継承の模式図

例題．以下は、学生の名簿を管理し表示するプログラムである。

```
// Person 抽象クラス
abstract class Person {
	protected String name;		// 名前
	protected int age;		// 年齢
	abstract void display();	// 表示メソッド
}

// Person クラスを継承した Student クラス
class Student extends Person {
	private String gakuseki_number;		// 学籍番号
	private static int student_num = 0;	// 全学生数

	// コンストラクタ
	Student(String name, int age, String gakuseki_number) {
		this.name = name;
		this.age = age;
		this.gakuseki_number = gakuseki_number;
		student_num ++;
	}
	void display() {
		System.out.println("Name: " +name + ", Age: " +age + ", Gakuseki: " + gakuseki_number);
	}
	static int getStudentNum() {
		return student_num;
	}

}

// main クラス
public class StudentList {
	public static void main(String args[]) {
		Student tanaka = new Student("Tanaka Taro", 21, "j13xxx");
		Student suzuki = new Student("Suzuki Jiro", 20, "j13yyy");
		tanaka.display();
		suzuki.display();
		System.out.println("Number of students: "+ Student.getStudentNum());
	}

}
```

```
/*  実行結果
Name: Tanaka Taro, Age: 21, Gakuseki: j13xxx
Name: Suzuki Jiro, Age: 20, Gakuseki: j13yyy
Number of students: 2
*/
```

プログラム仕様書作成課題

＜【Person】 のプログラム仕様書＞

作成者：J13000 情報太郎

（1）クラス
クラス名：Person
説明：人の情報を管理する抽象クラス
スーパークラス：なし

（2）フィールド
変数名：name
・属性：protected、非 static 変数、String 型
・フィールドの目的：人の名前を格納する変数

変数名：age
・属性：protected、非 static 変数、int 型
・フィールドの目的：人の年齢を格納する変数

（3）コンストラクタ：なし

（4）メソッド：
メソッド名：display
・属性：public、抽象メソッド
・戻り値の型：void 型
・引数：引数なし

＜【Student】 のプログラム仕様書＞

作成者：J13000 情報太郎

（1）クラス
クラス名：Student
説明：学生の名簿を管理するクラス
スーパークラス：Person

（2）フィールド
変数名：gakuseki_number
・属性：private、非 static 変数、String 型
・フィールドの目的：学籍番号を格納する変数

変数名：student_num
・属性：private、static 変数、int 型
・フィールドの目的：全学生数を格納する変数

（3）コンストラクタ
コンストラクタ名：Student
・引数：【引数名】、【型】、【変数の目的】

- name、String 型、学生の名前
- age、int 型、学生の年齢
- gakuseki_number、Sting 型、学生の学籍番号

・処理内容：
- 引数で受け取った学生の名前、年齢、学籍番号をオブジェクトの変数に格納する。
- 全学生数を表すクラス変数 student_num を 1 インクリメントする。

（4）メソッド
メソッド名：display
・属性：public、非 static メソッド
・戻り値の型：void 型
・戻り値の内容：戻り値なし
・引数：引数なし
・処理内容：学生の名前、年齢、学籍番号を表示する。
・オーバーライドの有無：有

メソッド名：getStudentNum
・属性：public、static メソッド
・戻り値の型：int 型
・戻り値の内容：全学生数
・引数：引数なし
・処理内容：全学生数を返す。
・オーバーライドの有無：無

＜【StudentList】 のプログラム仕様書＞

作成者：J13000 情報太郎

（1）クラス
クラス名：StudentList
説明：学生名簿に登録しし、内容を表示するクラス
スーパークラス：なし

（2）フィールド：なし

（3）コンストラクタ：なし

（4）メソッド
メソッド名：main
・属性：public、main メソッド
・戻り値の型：void 型
・戻り値の内容：戻り値なし
・引数：String
・引数：args[]、String 型、コマンドライン引数
・処理内容：
- 学生の名前、年齢、学籍番号を名簿に登録し、内容を表示する。
- 全学生数を表示する。

・オーバーライドの有無：無

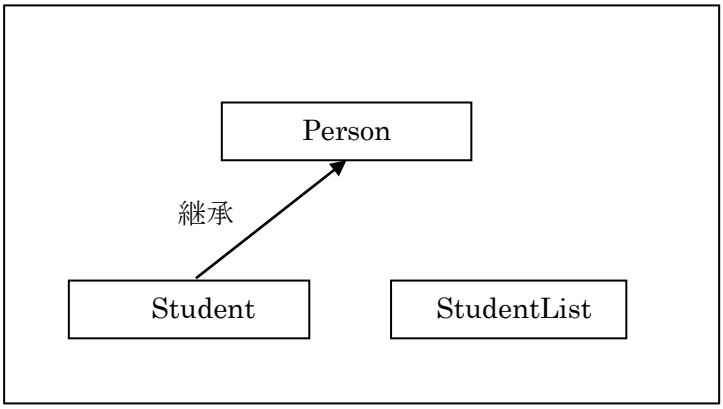

図 1. StudentList のクラス継承図