2. 簡単な java プログラム:StudentSample

2017/10/16

1 StudentSample

今回は、簡単な Java のプログラムを作成します。基本的な構造、クラスとメソッド、そして簡単なクラスの継承について学びます。

1.1 準備

前回作成したプロジェクト Introduction の下に、新しいパッケージ studentSample を作成します。その中に、三つの Java クラス、AbstractStudent、Student、そして StudentMain を作成します。StudentMain はメインクラスです。

1.2 クラスの継承

クラス AbstractStudent は、氏名 name と学籍番号 studentID という二つのフィールドを持っています。

課題 1 クラス AbstractStudent のコンストラクタの動作を説明しなさい。特に this の意味について注意しなさい。

クラス Student は、クラス AbstractStudent の継承クラスとして定義します。クラス AbstractStudent の二つのフィールドは、protected と指定されているため、クラス Student から利用することができます。また、二つのメソッドも public であるため利用することができます。

拡張したクラス Student では、加えた新しいフィールド record が定義され、その設

定及び取得メソッドが定義されています。

課題 2　二つのクラス AbstractStudent と Student の関係は理解できましたか? 特に、Student のコンストラクタの動作を説明しなさい。

1.3　泡立ち法

クラス Student には、成績に相当する record というフィールドがあります。このフィールドの値が小さい順に、クラスインスタンスを並べる (整列) することを考えましょう。そのための整列のアルゴリズムの一つとして泡立ち法 (bubble sort) を考えます。

```
for (i = n - 1; i > 0; i--) {
    for (j = 0; j < i; j++) {
        if ( A[j] > A[j+1] ){
            j 番目と j+1 番目の要素を入替
        }
    }
}
```

Algorithm 1.1: 大きさ n の配列 A に対する泡立ち法

クラス StudentMain は、main メソッドを持ちます。つまり、ここから実行することができるクラスです。main メソッドで実行している内容を見ていきましょう。

最初に、main メソッドでは、クラス Student のインスタンスを配列に収めます。次に、その配列をメソッド sort に渡し、その結果を印刷しています。

メソッド sort では、引数に渡されたクラス Student のインスタンスの配列を、record の値で小さい順に整列します。

課題 3　メソッド sort の中を実装しなさい。

課題 4　クラス StudentMain を実行し、正しく整列されていることを確かめなさい。

AbstractStudent.java

```java
package studentSample;

/**
 * 生徒の基本クラス
 *
 * @author tadaki
 */
public class AbstractStudent {

    //クラス内のフィールド
    //name と studentIDは一度定めると変更できない
    protected final String name; //名前
    protected final int studentID; //学生番号

    public AbstractStudent(String name, int studentID) {
        this.name = name;
        this.studentID = studentID;
    }

    //***** 取得メソッドと設定メソッド ********
    public int getStudentID() {
        return studentID;
    }

    public String getName() {
        return name;
    }
}
```

Student.java

```java
package studentSample;

/**
 * 生徒のクラス:StudentBaseの拡張
 *
 * @author tadaki
 */
public class Student extends AbstractStudent {

    private int record = 0;    //点数

    /**
     * @param name 名前
     * @param studentID 学生番号
     */
    public Student(String name, int studentID) {
        super(name, studentID);//親クラスのコンストラクタを利用
    }

    //***** 取得メソッドと設定メソッド ********
    public int getRecord() {
        return record;
    }

    public void setRecord(int record) {
        record = Math.max(0,record);
        record = Math.min(100,record);
        this.record = record;
    }

}
```

StudentMain.java

```
package studentSample;

/**
 *
 * @author tadaki
 */
public class StudentMain {

    /**
     * StudentRecordクラスを実行するためのmain
     *
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        //データの生成
        String names[] = {
            "Aoyama", "Asou", "Baba", "Edo", "Funaki",
            "Goto", "Gunji", "Ikeuchi", "Ito", "Mori"
        };
        int records[] = {90, 70, 88, 95, 100, 60, 45, 80, 95, 55};
        //長さnames.lengthのStudentクラスの配列
        Student students[] = new Student[names.length];
        //配列に登録、及び成績登録
        for (int i = 0; i < names.length; i++) {
            students[i] = new Student(names[i], i);
            students[i].setRecord(records[i]);
        }
        //整列の実行
        sort(students);
        //一覧を印刷
        for (int i = 0; i < students.length; i++) {
            Student s = students[i];
            String message = s.getName()
                    + "(" + s.getStudentID() + "):"
                    + s.getRecord();
            System.out.println(message);
        }
    }

    /**
     * ソートの実行
     *
     * @param array 対象となる配列
     */
    public static void sort(Student[] array) {
        int n = array.length;
```

StudentMain.java

この部分を実装

```
        }
    }
}
```