3P-2

非定型 Java 演習問題に対する自動採点システムの試作

小河原直行　山本富士男　宮崎　剛
神奈川工科大学　情報学部　情報工学科

１．はじめに

高等学校、大学、専門学校等、現在の教育機関においては教育内容にプログラミング言語の基礎学習が含まれることは少なくない。プログラミング言語の学習では、度々演習問題が出題される。その形式は「実際にプログラムを組む」というものが多く、これは対象者の習得レベルを比較的簡単に測ることができる反面、プログラミングの基礎とはいえ、採点に手間がかかる。

そこで、本稿ではオブジェクト指向のプログラミング言語として学習対象となることの多い Java 言語の演習問題に対する自動採点システムを、サーバサイド Java を用いて試作した。

２．自動採点システム

２．１．システム概要

自動採点には採点対象のソースファイル以外に、問題ごとに模範解答となるソースファイルと、採点の手順を記したテキストファイルが必要となる。教師側はこの二つのファイルを事前にシステムに送信しておくことになる。

自動採点システムは生徒側からソースファイルを受け取ると即座に採点を開始し、点数を返す。間違いがあればエラー項目として「メソッド○○の戻り値が一致しません」といった簡単なメッセージを表示する。採点結果はデータベースに格納され、教師、生徒はいつでも閲覧することができる。

２．２．採点の流れ

自動採点システムは生徒側からソースファイルを受け取ると外部プログラムとして Java コンパイラを動かし、ソースファイルをコンパイルする。これに加えて模範解答となるプログラムをロードし、採点対象、模範解答のそれぞれのインスタンスを生成する。その後、テキストファイルに記された採点手順に従い、それぞれのフィールド変数の値を取り出し、またはメソッドを実際に実行し、その戻り値を受け取る。そうして取得した両者の値を比較し、一致か不一致かを検証する。これをテキストファイルに記された数だけ繰り返し、一致ならば点数を加算していくことで最終的な得点を導き出す。また、不一致ならばその原因をエラーとして返す。

３．採点手順ファイル

３．１．採点手順ファイル概要

採点手順の書き方の例として、図１がある。

```
1    <Constructor>
2    (int 2);
3
4    <Object>
5    int[int 3] array;
6
7    <Process>
8    array[int 0] = int 3;
9    array[int 1] = int 2;
10   array[int 2] = int 1;
11   [15]num;
12   [30]sort(array);
13   [15]toString();
14   [20]Area();
15   num = int 5;
16   [20]Area();
17   java.lang.Math.PI;
```

図１．採点手順ファイル

採点手順は Constructor、Object、Process の三つのブロックから成る。Constructor ブロックにはインスタンス生成時に渡す引数、Object ブロックには採点作業で使用したいが、採点対象のプログラムにはないオブジェクトを作成するための文をそれぞれ記す。最後の Process ブロックに、実際の採点手順を記すことになる。

３．２．採点項目の書式

採点対象のプログラムに組み込まれたフィールド変数名やメソッド名を記述することにより、それらフィールド変数の値やメソッドの戻り値を自動採点システムに検証させることができる。メソッドに関してはメソッド名のあとに引数を括弧で括る必要があり、またその場合、引数の直前に「Object」と記述することで、引数を Object 型として渡すことができる。

５行目では、最後に「array」という名前を指定しオブジェクトを生成しており、これは Object

ブロックにおいてだけ有効な書き方である。

８、９、１０行目のように、プリミティブ型のクラス名のあとに数値等を記述することで、その数値を指定したプリミティブ型として扱うことができる。例外として String 型もこれに含まれ、float 型と double 型を加えた以上三つの型に関しては、値を「"」で括る必要がある。

１７行目にあるように、パッケージも含めたクラス名のあとに続けることで、static なフィールド変数やメソッドを呼び出すこともできる。「=」を挟むことで、値の代入もできる。

検証の結果、一致していた場合に加算する点数は、それぞれの行の先頭で１１、１２行目等のように括弧で括って指定することになる。点数を指定していない行に関しては検証を行わない。

４．採点プログラム

４．１．処理の流れ

採点プログラムは採点手順ファイルの文字列をトークン単位で読み込み、その値によって、インスタンスの生成、フィールド変数、メソッドの呼び出しといった処理を適切に行っていく。その結果、採点手順の一行ごとの値を取得し、それを検証する。以下に、各処理の流れを説明する。

４．２．インスタンス生成

読み込んだトークンがパッケージも含んだクラス名だと判断された場合、採点プログラムは新規にインスタンスを生成する。

まず、Class クラスの forName メソッドを利用し、Class オブジェクトを生成する。次に同クラスの getConstructor メソッドに適切な引数を渡して Constructor オブジェクトを生成し、Constructor クラスの newInstance メソッドで新規インスタンスを生成する。

４．３．プリミティブ型または String 型の生成

読み込んだトークンがプリミティブ型のクラス名、または String だった場合、採点プログラムは新規にプリミティブ型、または String 型を生成する。Integer クラス、Double クラス、String クラス等を利用する。

４．４．フィールド変数等の値の取得

読み込んだトークンがフィールド変数名、または Object ブロックで生成したオブジェクト名だと判断された場合、採点プログラムはそこに格納されている値を取り出す。

フィールド変数ならば、Class クラスの getField メソッドを利用し、対象となるフィールド変数を持つ Class オブジェクトから Field オブジェクトを生成する。次に Field クラスの get メソッドで値を取り出す。

Object ブロックで生成したオブジェクトならば、生成時にハッシュテーブルへオブジェクト名をキーとしてマップしているので、Hashtable クラスの get メソッドを利用し、値を取り出す。

４．５．メソッドの呼び出し

読み込んだトークンがメソッド名だと判断された場合、採点プログラムはそのメソッドを呼び出す。まず、Class クラスの getMethod メソッドに適切な引数を渡して、対象となるメソッドを持つ Class オブジェクトから Method オブジェクトを生成する。次に Method クラスの invoke メソッドに適切な引数を渡して、メソッドを呼び出す。

５．実験・結果

一例として、図１の採点手順ファイルと図２の模範解答ソースファイルを登録した。例えば、生徒の解答において、図２の２６行目の”num*num”が”2*num”のように誤っていた場合、即座に、”80点：メソッド Area の戻り値が一致しません”という応答を生徒に返した。

```
1   public class Question {
2       public int num;
3
4       public Question(int num) {
5           this.num = num;
6       }
7
8       public int[] sort(int[] number) {
9           for(int i = 0; i < number.length - 1; i++) {
10              for(int j = number.length - 1; j > i; j--) {
11                  if(number[j] < number[j-1]) {
12                      int temp = number[j];
13                      number[j] = number[j-1];
14                      number[j-1] = temp;
15                  }
16              }
17          }
18          return number;
19      }
20
21      public String toString() {
22          return Integer.toString(num);
23      }
24
25      public double Area() {
26          return Math.PI * num * num;
27      }
28  }
```

図２．模範解答ソースファイル