研究論文

プログラミング演習における評価方法の改善

A New Programming Exercise Evaluation Method for Programming Exercise Lesson

松浦佐江子
芝浦工業大学システム工学部

Abstract: Mastering programming skills is an indispensable requirement for education in the field of information technology. If a student does not fully learn the basics of programming through a programming exercise lesson, it will be an obstacle in learning practical software development technology. A student can develop an in-depth understanding of a program not only by solving a problem but also by testing the program or explaining its objective. An evaluation method that uses the test program and code review process was adopted in a programming exercise lesson in order to improve the students 'programming skills and motivate them to learn. We are also developing an online lesson support system that will provide the students with various types of support for lessons. In this paper, using the lesson support system, we propose an evaluation method that was adopted for the programming exercise lesson.
Keywords: code review, programming exercise, lesson support system, software engineering education.

1．はじめに

情報技術教育においてプログラミング技術の習熟は必須事項である．プログラミング教育では文法・基本的なアルゴリズムの定義・設計の方法を講義し，演習によって，プログラミング技術を訓練する．プログラミング演習の実施方法は教員が提示した課題を学生が授業時間に解いて提出し，提出したプログラムを教員が評価するという方法が一般的である．授業時に教員やTAが質問に答えることはあるが，評価は提出されたプログラムのみから行われる．演習レベルのプログラムはサンプルとして書物やインターネット上で公開されていることが多いので，学生が自分で課題を解かずに，見つけたサンプルのコピーをレポートとして提出していることが多い．良いサンプルをお手本にプログラミングを覚えることは技術を磨く上で重要ではあるが，コピーを単純に評価していたのでは，プログラミング技術を習得しているか否かを適切に評価することはできない．また，学生もコピーしただけのレポートを他人が提出している状況を見て，学習意欲が削がれることがある．われわれは，プログラミング演習レポートを直接対話によるコードレビューでの審査方式で評価することにより，学生がどのように考えてプログラミングしたかを評価し，学生の学習意欲の向上を図る工夫を試みた．本稿ではこの評価方式について報告する．

2．プログラミング教育の問題点

本学科におけるプログラミング言語教育ではC言語とJava言語の教育を行っている．プログラミング演習の目標は，要求を満たすプログラムを作成することである．文法と基本的なプログラミング技術を習得した後には，要求を満たすプログラムを作成するにはどのように問題を分析し，プログラムを設計したらよいか，問題に対してなぜプログラムはこの構成がよいのか，要求を満たしていることをどのように確認するのかといった事柄を学習しなければ，プログラミング技術を習得したとは言えない．われわれのプログラミング教育の最終目標は，グループである程度の規模を持つソフトウェアを開発する実習により，

Saeko Matsuura
Shibaura Institute of Technology
E-mail:matsuura@se.shibaura-it.ac.jp

（受付：2005年7月2日，受理：2005年10月1日）

プログラミング技術とプロジェクトマネージメント技術を学習することである[1]．プログラミング学習支援には，e-Learningによる自習方式[2][3]，ソースコードの公開による学習[4]，小テスト形式で予習・復習する[5][6]といった自ら積極的に学習することを期待する方式が多く実施されている．このような学習の機会を豊富に提供することに加えて，個々の学生の理解度を適切に把握して妥当な評価を行うことも，全体の学習意欲の向上には必要である．

３．プログラミング技術評価の問題点

前述のプログラミング演習の一般的な実施方法における問題点は以下の通りである．

① プログラムの正当性を判定するために，教員自身がプログラムをコンパイル・実行しなければならない．１回に100程度のプログラムが提出されるため，個別にコンパイル・実行し，条件を満たしているか否かを逐一評価するのに非常に手間がかかる．

② Hello Worldのような非常に単純なプログラムの場合にはレポートの差異は生じないが，ある程度の複雑度の問題であれば，複数の解法が存在する．類似，あるいはまったく同じ解答が多数存在するが，どの解答がオリジナルであるかを判別することはできない．サンプルコードのコピーであっても，評価においてはそのプログラムの内容のみが評価されることになり，個人の理解や努力とは無関係である．

③ 類似の解答が多数存在する場合，あらかじめパターンに分類しておかないと，均一な評価が難しくなる．

④ 記述の意図を読み取ることが困難である．コメントが付加されている解答もあるが，変数や場合分け等の状況の説明であり，なぜ，このようにプログラムを記述したかの説明ではない．

これらの問題を解決するために，われわれはコードレビューと自動採点可能なテストプログラムを用いたプログラミング演習レポートの評価方法を提案する．本方法の基本的なアイデアは以下の通りである．

① 授業支援システムのレポート管理機能を利用してプログラムを提出させ，コンパイル・実行を可能とする．

② 直接対話によるコードレビューにより，学生の理解度を判定するための審査方法を策定する．

③ 審査において基本的なテストプログラムを用意し，テストを自動化して効率よく均一に評価を行う．以下，基本的なテストを自動採点するプログラムを「自動採点プログラム」と呼ぶ．

４．プログラミング技術評価方式の提案

本方式を２年生の後期に行われるJavaを用いたプログラミング演習に適用した．この段階における演習の目的は，教員が問題提示時にインタフェースに関する仕様を与え，その意味を学生が理解し，オブジェクト指向の概念を学習することである．

以下，本方式による評価を「審査」とし，従来のように教員がレポートを閲覧・コンパイル・採点し，コメントを付加する評価を「採点」と呼ぶことにする．

コードレビューによる審査は，一定時間内に多人数に対して公平に実施しなければならない．これらのことを考慮して，プログラミング演習の評価を次のように行った．

(1) 授業支援システム

われわれはe-Learningを支援するために授業支援システムを研究開発し，2003年度後期より運営している[7][8]．本システムの次の機能を利用して，プログラミング演習のレポート管理ならびに評価を行う．

① レポートを期限までに電子的に提出する．図１に示すように，パッケージを用いたJavaのプログラムも複数ファイルとしてそのまま提出することができる[8]．

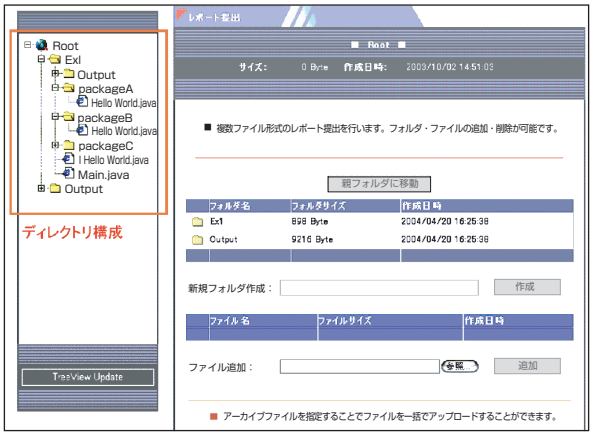

図１　レポートの提出

② 評価時に学生が提出したファイルを自分でダウンロードする．

③ 教材として自動採点プログラムを配付する．

(2) 自動採点プログラム

効率よく評価を行うためには，問題とその評価方法を組み合わせて考案する必要がある．本対象授業では，プログラムの構成に関する条件や，メソッドの仕様を与えて，それらを満足するようにプログラムを作成することを達成目標の一つとしている．このことから，課題のプログラムの構成に関する制約を予め与え，以下の項目に関して自動採点プログラムを用意する．

① 構成テスト

課題に指定したプログラム構成（クラス，フィールド，メソッドのシグネチャ，コンストラクタ）を満たしているかをテストする．

② 機能テスト

要求された機能を満たしているかを指定したクラス・メソッドに基づきテストする．

自動採点プログラムはリフレクションを用いて定義されており，Javaのclassファイルとして提供される．実行すると，成功（○）または失敗（×）がテスト結果として標準出力に表示される．

(3) コードレビューによる審査

直接対話によるコードレビューにおいては以下の内容を口頭で質問し，学生の返答を５段階で評価する．

① 変更テスト

口頭でメソッドに対する変更要求を伝え，プログラムのどこを，どのように変更しているかを観察する．

② 基本概念のテスト

リングバッファやスレッドの同期といった各課題における基本概念についてプログラムを見ながら説明させる．

一つの課題に対するコードレビューは公平を期すために１回の授業時間（90分）内で実施しなければならない．また，質問を十分に行えるように一人の審査時間をある程度確保する必要がある．対象人数が多人数（60名から100名）であることから，以下のように審査要領を設定し，効率よく審査を実施する．

① 審査員は教員２名およびTA６から７名とする．

② 主たる審査員が自動採点プログラムと上述の口頭質問による２種類のテスト項目およびその評価基準を示した評価リストを他の審査員に提示する．各審査員は，この評価リストに従って，自動採点プログラムを利用してコードレビューを行う．評価項目数は問題によって多少異なるが，20項目程度である．

③ １回の審査は授業時間（90分）内に行う．

④ １人の審査時間は８分から９分とする．

⑤ 学生には課題配付時に指定日に審査を行うことを予告する．

⑥ 審査中は席を離れてはいけないことを学生に通達する．

審査の手順は以下の通りである．

① 学生は，提出したレポートを審査員の前でダウンロードする．これで，期限までに提出したレポートによる審査が実現できる．

② 学生は，ダウンロードしたプログラムをコンパイル・実行する．
③ 学生はシステムから自動採点プログラムをダウンロードし，適切な位置に配置し，テストを実施する．自動採点プログラムを実行すると，予め審査員に示されたテスト項目とテスト結果（○×）が表示されるので，審査員はこれを記録する．
④ 審査員はあらかじめ決められた質問項目を質問し，ソースコードを閲覧しながら学生に説明させる．この説明を評価基準に従い５段階で評価する．

５．考察

(1) 演習の結果

2003年度および2004年度のプログラミング演習において，７回のレポート提出のうち，３回を審査方式で，４回を通常の採点方式で採点を行った．審査を受けた学生数は2003年度が平均106名（履修者の66%），2004年度が57名（履修者の53%）である．

2004年度演習終了時にアンケートを実施した（回答数48名）．「審査方式は従来の採点方式に比べてよいと思うか」という問いに対して，「強くそう思う」23%，「そう思う」56%，「どちらともいえない」17%，「そう思わない」４%，「まったくそう思わない」０%という結果になり，学生の79%（強くそう思う，そう思う）が審査方式を支持している．支持している理由は以下の通りである．

- 会話することでプログラムの意図を伝えることができる．
- 自分の理解度を確認でき，理解力が深まる．
- 実際にプログラミングした人にしかわからない質問があり，「自力でつくったか」「内容を理解しているか」を評価してもらえるのがよい．
- その場で評価結果がわかるのでよい．
- 他人にコピーを提出している人と自力で解いた人の差がつけられるのが良い．
- 提出に緊張感が出た．

まれにではあるが，方法がわからないためにコンパイルやプログラムの実行ができない学生がいた．他人のプログラムをそのまま提出して自分でプログラムを作成したことがないからであるが，通常の評価ではこの事実はわからない．

2003年度には自動採点プログラムを用意しなかったため，課題の条件チェックに時間がかかるという問題があった．2004年度は自動採点プログラムの使用により，課題の条件チェックや，基本的なテストケースによるテストの自動化が可能になった．これにより，授業時間内に効率よくテストを実施することができた．

(2) 評価方法の問題点

審査方式の問題点として，次の点に関して，不公平を感じているという意見が数件あった．

- 審査員が教員であるかTAであるかによって質問の仕方が違っていた．
- 質問が隣の人に聞こえた．

実施上の問題点としては，評価者に依存しない審査の実現がある．運用上学生一人の審査時間を８～９分確保するためには，審査員の人数をある程度確保する必要がある．本演習ではJavaプログラミングに関しては一定水準の技量を持っている大学院生および学部4年生をTAとして採用した．事前に評価基準に関する注意を与えているが，質問の仕方についてさらに訓練をする必要がある．

(3) 評価結果の分析

2003年度は160名の履修登録に対し140名が，2004年度は107名の履修登録に対し94名が１回以上レポートを提出した．図２は2003年度の，図３は2004年度の履修学生の審査平均点と採点平均点の散布図である．審査平均点および採点平均点が60点以上の学生は７回のうち５回以上レポートを提出し，2003年度は点数に正の相関関係（相関係数は0.73）が見られる．

その他の学生は，結果的にはレポート提出回数も少なく，単位を取得することができなかった．１回以上レポートを提出し，審査には参加していない学生が，2003年度には13名，2004年度には25名いた．個別の調査は実施していないが，彼らは他の学生のコピー等をレポートとして提出しているために，自分自身で答えなければならない審査を受けることができなかったものと思われる．

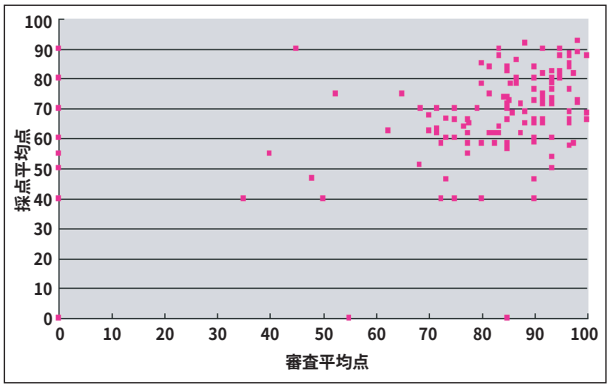

図2　2003年度評価結果

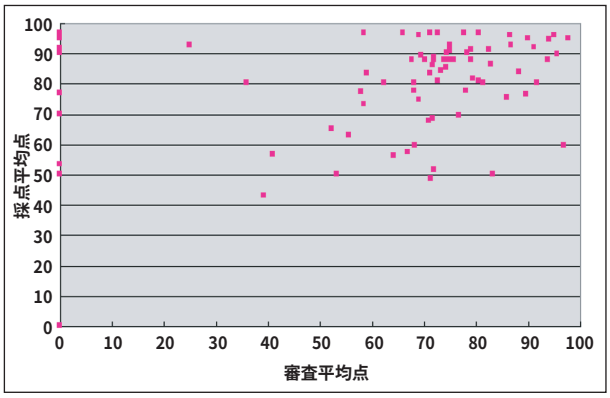

図3　2004年度評価結果

6．まとめ

本方式で次のような効果を得ることができる．

① 他人のプログラムをコピーして提出している場合でも，質問により，個人の理解度を判定することができる．

② 構成テストを行うことで，学生にプログラムの仕様の重要性を認識させることができる．学生はテストで×がつくことに敏感である．何人かの学生は，プログラムの修正後に自動採点プログラムを適用して確認していた．

③ アンケートの意見にも見られるように，自己の努力に対して妥当な評価がされることによる学習意欲の向上が見受けられる．

本稿で示した方法により，学生は自動採点プログラムを用いたテストの方法を学習し，プログラムを説明することを経験できる．学生にも好評であるので，課題とその自動採点プログラム，質問と判断基準を総合的に考案し，審査方式を実施していきたい．また，提出の前にプログラムを自動採点プログラムで確認したいという意見もあり，自動採点プログラムの配布は学生の自習を促すことになると考えられることから，その運用を検討している．

参考文献

[1] 松浦佐江子,青沼俊介,吉田明広:グループワーク支援システムを用いた実践的ソフトウェア工学教育. 情報処理学会FIT2004情報科学技術レターズ, Vol.3, pp.351-354, 2004.

[2] C. Daly and J. M. Horgan : An Automated Learning System for Java Programming. IEEE Transaction on Education, Vol.47, No.1, pp.10-17,2004.

[3] H.Mungunsukh and Z. Cheng:An agent based programming language learning support system.International Conference on Proceedings of Computers in Education, pp.148-152, 2002.

[4] Crew (Creative Workspace): http://www.crew.sfc.keio.ac.jp/

[5] CEAS (Web-Based Coordinated Education Activation System) : http://ceasdemo.iecs.kansai-u.ac.jp/

[6] WebCT : http://www.webct.com/

[7] S.Matsuura and S.Atsuta : Lesson Support System for Programming Exercise Lesson Improvement, Proc. of The 4th IASTED International Conference on Web-Based Education (WBE2005), pp.131-136, 2005.

[8] 熱田智士, 松浦佐江子:Javaプログラミング演習向け課題レポート提出・管理機能を付加した授業支援システム. 情報処理学会FIT2004情報科学技術レターズ, Vol.3, pp.359-362, 2004.