プログラミングの写経型学習過程を対象としたつまずきの分析とテキスト教材の改善
―作業の自立的遂行と作業を介した理解のための支援と工夫―

岡　本　雅　子
（京都大学大学院情報学研究科）
村　上　正　行
（京都外国語大学マルチメディア教育研究センター）
吉　川　直　人
（株式会社キヤミー）
喜　多　　一
（京都大学国際高等教育院）

Analysis of Missteps in *Shakyo*-Style Learning of Computer Programming and Improvement of Learning Material
—Support and Design for Self-Sustaining Work Performance and Understanding through Work—

Masako Okamoto
(Graduate School of Informatics, Kyoto University)
Masayuki Murakami
(Research Center for Multi-Media Education, Kyoto University of Foreign Studies)
Naoto Yoshikawa
(CAMMY, Co., Ltd)
Hajime Kita
(Institute for Liberal Arts and Sciences, Kyoto University)

Summary

This paper defines "*shakyo*-style learning" as the learning of computer programming by mimicking input according to sample programs, running them and ensuring their outcome. We developed *shakyo*-style learning material based on missteps found in the process used by novices in learning computer programming; we then evaluated it in an undergraduate computer literacy course. As a result, we found that learning material with documented in-detail work instructions for the programming process worked well among novices. Additionally, through smoothed work in learning, this learning process had strong implications for understanding the concepts in computer programming.

キーワード：プログラミング教材、写経型学習、つまずき、事例学習、初学者
Keywords: computer programming materials, *shakyo*-style learning, missteps, example-based learning, novices

1. はじめに

コンピュータプログラミングの学習では、一般に、学習者は、提示されたサンプルプログラムを模倣して入力、実行し、その結果を確認する過程を経る。特に、入門段階にある学習者にとっては、他の方法を併せ用いるとしても、この段階を踏まずしてプログラミング技術を身につけることは難しいと著者らは考える。実際、ほとんどの初学者を

対象としたテキスト教材（例えば、柴田、2004；高橋、2012）では、プログラムの概念的、理論的説明のほかに、サンプルプログラムを読むこと、実行結果の画面キャプチャ（画像）を見ること、そして最後に、実際にプログラムを打ち込んで動作させるといった作業を要求している。単にプログラムの文法や命令の機能を理解させたいだけであれば、この学習過程のうち、「概念的説明と実際のサンプルプログラムを読む」ほか、「実行結果の画面キャプチャを見る」程度で十分であり、「実際にプログラムを打ち込んで動作を確認する」必要はない。それにも関わらず、あえて命令を入力させ、動作させ、その結果を確認させるのは、プログラミング学習が書いてあるプログラムの意味を理解するだけではなく、目的とするプログラムを組み、そして実際に動作させることが出来るまでのスキルの習得を目的としており、このような演習過程に学習効果が期待できるためであると考えられる。

しかしながら、プログラミング学習におけるこのような演習過程の必要性や具体的使用の方略については、これまで学術的な研究の対象とされておらず、当然ながら、これらに関する報告は見当たらない。また、演習過程を重視するテキスト教材（例えば、河西、2008）が数多く出版されており、各著者が各々その方略の有効性やテキスト教材の全体の構成やレイアウト等の編集時の工夫を述べてはいるものの、学術的な根拠までは示されていない。このように、「プログラムを模倣して実行する」といった学習方略については、それがごく一般的（教える側の暗黙上の共通認識）であるため深く省みられてこなかったものと推察されるが、それゆえまた、プログラミング学習において根源的でかつ一般性の高い研究対象であると著者らは考える。

以上を踏まえて、本研究では、初学者を対象としたプログラミングの演習過程のうち、サンプルプログラムの模倣から始めて、学習者自身がサンプルプログラムにある命令、文法などを一部改変したり、組み合わせたりするなどして試していくまでの過程を「写経型学習」と規定し、まずこの「写経型学習」を中心とした学習過程で発生する学習者側の問題点を抽出することとした。続いて、そこから得られた知見をもとに「写経型学習教材」を開発し、2009年度の京都大学の全学共通教育科目「情報科学演習」において検証を行い、2011年度の同科目においてさらにいくつかの改善を加えて再度開発した教材の有効性を検証した。

2. 関連研究

2.1 概念そのものの学習と事例学習（事例からの学習）

学習課題となる概念の習得に向けた具体的な方法としては、大別して、「概念そのものの学習」、「事例学習（事例からの学習）」の2つが挙げられる。これら2つの方法は、目標とするところは同じであるが、どちらを重視するか、もしくは、学習の導入段階においてどちらから始めるべきかといったところにおいては議論が分かれるところである。このような中で、「概念そのものの学習」から始める方略の優位性を示す事例がいくつか報告されている。

まず、数学教育の分野では、具体的事例を用いた学習それ自体に困難性が潜んでいるという見解が示されており、Kaminskiら（2008）は「具体的事例から抽象的要素を抽出することが難しい」と指摘している。Kaminskiらは、大学生を対象として、具体例を用いて数学的な概念を学んだ学生と抽象概念から直接学び始めた学生を比較した際、具体例を学んだ学生は新しい状況にその知識を応用できなかったのに対し、抽象的な記号などを用いて同じ概念を学んだ学生は、異なる状況に応用できる場合が多かったと報告している。このように、Kaminskiらの報告では、「概念そのものの学習から始める方略」の優位性を示す一方で、具体的事例から始める学習方略の困難性を指摘している。

プログラミング教育分野では、長谷川ら（1997）の報告に「抽象概念（イメージ）の習得から学習を始める方略」の有効性が示されているが、こちらは、事例学習の困難性については言及しておらず、その点において文脈が異なる。

長谷川らは、プログラムの動作を条件によって切り替えたり、同じ動作を繰り返し行うなどの「制御構造」について、そのイメージを自由に描画によって表現する方法で、学習者のイメージ（二次元抽象図）を調査した結果、プログラミングを習得するには、プログラム実行時の処理の流れの明確なイメージを持っている、つまり、実行時の動きを抽象化してとらえたイメージを持っている学習者ほどプログラミングの理解度が高いことを示した。すなわち、初学者を対象にプログラミング教育を行う場合、プログラムの実行時のイメージを持たせることが効果的であるという指摘である。さらに、長谷川らは、プログラミング初学者を対象とした教育において、制御構造を表す図を提示し、処理の流れのイメージを形成することによって理解を支援することにより、学習効果が得られることを明らかにしている（長谷川・山住、1998）。

これら長谷川らの一連の実践例では、イメージを図として示しているため、一見、理論や概念といった文言となじまないように思えるが、模倣を繰り返すことによる事例学習のなかで処理のイメージを構築させるのではなく、学習者が処理のイメージあるいは抽象概念を習得した後に実習に入っている点において、概念学習先行型の方略とみなすことができる。

なお、本研究は、「事例学習」の要素を含む「写経型学習」を対象としているが、同方略の問題点の抽出とそれを実施する際の具体的方法の提示を目的としており、どちらを重視するかあるいは先行して教授すべきかについては議論の対象としない。

2.2 プログラミング学習とつまずき箇所

プログラミングの初学者は、学習の初期段階でつまずくことが多い。とりわけ学習の初期段階では、プログラムのタイプミスに起因する文法エラーが多発する。ここでは、コンパイル（作成したプログラムをコンピュータが実行可能な形式に変換する）操作の際に、タイプミスを示すメッセージが専門用語を含む英語で表示されるため、初学者にとっては、用語、内容ともに初見であり、その意味を理解するのが困難である。さらに次の段階として、自分が作成したプログラムの実行結果が予想していた結果と異なるということもある。ここでは、先述した文法エラーとは異なり、エラーメッセージが表示されないので、学習者が間違っている箇所を見つけられないことが多い。このような文法エラーや実行時のエラー箇所を自分で発見できずデバッグ（プログラムの誤りを探して修正する作業）ができないなどのつまずきについては、江木・竹内（2009）、そして太田（2009）などによって指摘されている。また、「プログラミングの基本である順次処理（プログラムの命令を上から順番に実行すること）を理解することも、初学者には難しい場合がある」（岡本ら、2010）、「条件式をどのようにつくればよいのかわからずに、ここで理解できなくなる学生も多い」（深町、2010）、そして「for 文の入れ子的な利用も初学者が典型的に困難さを抱えるものである」（阿部、2009）などの指摘がある。また、河内谷（2006）は、for 文の学習は認知的負荷が大きいとして、「類似概念をこれまでに目にしたことがない」こと、「多くのデータから判断を下す、かなり複雑な問題」であることがその主たる要因ではないかと考察している。

3. 2009 年度の実践

3.1 企業研修の写経型学習過程におけるつまずきの抽出

本研究では、写経型学習過程における問題点の抽出を行うこと、それに基づいて作成したテキスト教材を検証すること、これらを大学の授業内で行いたいと考えていたが、問題点の抽出に際し、詳細な調査を実施する場が見当たらなかった。こうした折、著者らは、初級プログラマ養成研修を実施している企業を対象に、研修の実施状況を参与観察および研修修了者と講師を対象としたインタビュー調査を行う機会が得られたため、ここでみられる写経型学習を中心とした学習過程で発生する学習者のつまずき箇所を抽出することとした。なお、同企業では、写経型学習を重視した初級プログラマの研修を実施しており、テキスト教材の説明部分を読んで理解する過程に加え、サンプルプログラムを改変して試す過程についても個別学習に委ねており、各学習過程において不明な点や自ら解決できない箇所についてのみ講師に質問するように指示していた。

同企業における調査結果の分析をもとにテキスト教材を作成し、大学において検証を行うことについては、対象の場や対象となる学習者層の相違が問題となるが、高校卒業以上のプログラミング初学者を対象としている点では大学の授業における受講生と条件が一致しており、また、研修参加者が 3 人から 5 人程度と詳細な質的調査を行うことが可能な人数であったことから、著者らは本研究の特性に鑑みて、予備調査の対象として適切であると判断した。

企業研修の参与観察およびインタビュー調査から抽出されたつまずきのうち、特に写経型学習に係るものについては、以下の 2 つに類型化することができた。

(1) 写経型学習を遂行する上で自立的に作業することができない

事例 1）コンパイルの手順を覚えられない、どの手順から進めてよいのかわからない

事例 2）エラー発生時に修正が必要な箇所を発見することができない

(2) 写経型学習の過程から目的とする内容を学び取ることができない

事例 1）学習当初は、実際に入力したプログラムの内容と画面に表示される実行結果が頭の中で関連付けができていなかったため、何を学んでいるのか理解できない

事例 2）条件によって繰り返し処理が行われていることが認識できなかった。また、なぜ繰り返し処理が必要であるのかが理解できない（繰り返し処理を用いる理由や目的が理解できない）

すなわち、これらの事例において見られたのは、作業遂行上のつまずきと作業を介した理解に係るつまずきであった。なお、同企業の調査の結果、明らかになったつまずきについては、「初学者を対象とした自習中心のプログラミング教育の教材開発と評価」（岡本ら、2010）において既報である。

「写経型学習」は、学習者が自ら入力し実行する「作業遂行の自立性」と、結果を確認し、プログラムの記述と結果の関係から自ら学び取っていく「作業を介した理解」の 2 つの側面に特徴がある。そのため、これら 2 つの問題点を改善によって、作業遂行におけるつまずきをなくすこと、また、実行結果を見て、学習者が命令と動作の因果関係について理解できるようにすることが学習を効果的に進めるための必要条件であると考えられる。

作業を介した理解を促進するためには、具体的作業から獲得すべき抽象概念や動作のイメージなどを容易に掴み取れるようにする必要があり、具体的作業における不要な認知的負荷の軽減が求められる。そのためには、具体的作業そのものを、抽象概念の抽出やイメージの形成に適したものに改善する方法も考えられるし、事前に習得すべき抽象概念などを示しておいて、後の具体的作業でそれを実感させるといった方法も考えられる。これらの方略のうち、前者については、扇風機と類似の形状を持ち、またその機能を模倣できるマイコンボードを用いて直感的に理解できるよう工夫した組み込み系 C 言語プログラミング教材を開発し、京都大学の全学共通科目「情報科学演習」において 2010 年度の前期に実践と検証を試みた際、それぞれの命令に対応する個別の動作に注意を向けることができたという結果が得られており、このような認知的効果が示唆されるなど一定の効果が確認されている。その内容については、「『視覚的顕在化』に着目したプログラミング学習教材の開発と評価」（岡本ら、2013）において既報である。しかしながら、同論文で示した方法は、特殊な装置を用意することとそのための特殊なプログラミング環境を用いる必要があり、一般的なプログラミング教育課程で用いようとすると、教授者側の負担も相応に大きくなるものとも考えられる。つまり、同報告で使用した方略には汎用性といった側面において問題があり、より広く用いることが出来る方略についても実践的に検証していくべきであると著者らは考えた。

以上を踏まえて、本研究においては、特別な学習用の機材等を用意せずに、従来のテキスト教材の利用を前提とした汎用性の高い改善方法について提示したいと考え、後者の方略を採用して教材の開発と実践を試みることとした。

3.2　写経型学習教材の開発

プログラミング初学者を対象としたテキスト教材では、サンプルとして示したプログラムが正しく動くことが前提となっており、エラーが起こったときの対処まで記述しているものは少ない。また、プログラミング言語の紹介に重点を置いたテキスト教材では、文法の解説に必要な部分だけを例示するような記述がしばしば見受けられるが、プログラムの動作を確認するためにはプログラム全体を提示する必要がある。このため、初学者がテキスト教材を用いて学習した際に、一か所でもつまずいてしまうとその時点で学習を諦めてしまうことやエラー対応などに時間を費やしてしまう場合がある。そのため、本研究では、3.1 節で述べたプログラミング学習における学習者のつまずきを踏まえ、以下の 2 つの方針に基づいて教材を作成し（表 1 参照）、同テキスト教材を 2009 年度の後期に京都大学で開講された「情報科学演習」で使用した。テキスト教材の開発においては、自立的な作業遂行のために「コンパイルの手順やエラー発見の方法についてその手順を掲載すること」と、作業を介した理解のために「学習の目的、学習の仕方など、学ぶ内容とともに学び方を明示すること」という 2 つの方略によって問題解決を図った。なお、本研究では、プログラミング言語の文法面の網羅的な紹介ではなく、学習者が実際に手を動かしてプログラミングを実践することによって理解を促すことに重点を置いており、そのような学習の際に必要とされる改善を試みている。

表１　開発したテキスト教材の目次

目次	例題数
学習の仕方を理解する	—
プログラミングに使用する文字と用語	—
C 言語プログラムの開発手順	—
プログラムを作成してみよう	8
画面に表示する方法	5
データを入力する方法	2
値を保存する方法（変数）	8
処理の流れとフローチャートを理解する	—
条件によって処理を分ける方法（if 文）	6
変数を一括管理する方法（配列）と繰り返し処理（for 文）	11
配列の便利な使い方	2

(1) 自立的な作業遂行のための支援

コンパイル時のつまずきを軽減するため、エラーの原因となる「見えない文字（空白）の使用」や「大文字と小文字の区別」、「半角全角の区別」、「カッコのとじ忘れ」などについて、本文中に注意書きを添えたほか、エラーメッセージの実例を別紙で 2 例紹介した。

本研究では、初学者が自立的に作業を遂行することを考慮して、サンプルプログラムは「完全に動作するプログラム」と「その実行結果」を掲載する方針を取った。そして、サンプルプログラムを対象に、プログラムの各行に対応して逐次解説を加えたほか、プログラムの構成要素の紹介も併せて行った。なお、前述の方針から、プログラミング言語の文法の詳細は他のテキスト教材等を参照させるものとし、実践的に利用するポイントに焦点を当てて解説した。

(2) 作業を介した理解のための支援

本テキスト教材では、各章のはじめに「習得・演習目安時間」を提示するとともに、その章では何を理解しなければならないのかを「学習の目的」として、2 から 3 項目ずつ明示した。たとえば、第 6 章の「値を保存する方法（変数）」では、「値を格納する箱である『変数』について理解する」、「変数に値を格納する方法（代入）を理解する」としている。このように、各章に「学習の目的」を説明しただけでなく、これとは別に、「学習の仕方を理解する」という章を設け、本テキスト教材を使用してプログラミングを学ぶための手順を示した。

また、プログラミングの学習が進んでいくにつれ、それまでに学習した既出項目を活用する機会が多くなり、学習者は、新たに学ぶ項目とそれまでに学習した項目の両方でつまずくことが懸念される。そのため、学習者がなるべく失敗しないように学習ステップを細かく設定し（スモールステップの原理）、1 つのサンプルプログラムから新たに学ぶ項目は 2 つ以内に絞って提示した（日本教育工学会、2000）。さらに、本テキスト教材では、初学者にとって難しいとされる繰り返し命令 for 文について「for 文ドリル」という章を設け、それまでに学習した内容を組み合わせて活用した例題を数多く提示した。ここでは、プログラムの構造の理解を促すため、サンプルプログラムの内容を少しずつ改変させた例題を用意した。

3.3　開発した教材を用いた授業実践の対象と方法

3.3.1　対象とした授業

本研究の対象とした授業は、2009 年度の後期に京都大学で実施された全学共通科目（全学部の 1 回生から 4 回生まで受講可）「情報科学演習」である。本科目は、教養教育におけるコンピュータリテラシ教育という位置付けで週 1 回 90 分の選択科目として実施したもので、全体の構成は、Word による文書整形、PowerPoint と Excel の基本、HTML、CSS の基本に加え、C 言語によるプログラミングに後半 5 回の授業を割り当てている。5 回に渡るプログラミングの授業は、プログラムの論理構造の理解を目標としており、プログラミング体験を通して「コンピュータの動

表 2　授業内容

授業内容	2009 年度の内容	2011 年度の内容
1 回目	C 言語プログラムの開発手順 プログラムを作成してみよう	C 言語プログラムの開発手順 プログラムを作成してみよう
2 回目	画面に表示する方法 データを入力する方法	画面に表示する方法 データを入力する方法 値を保存する方法（変数）
3 回目	値を保存する方法（変数）	条件によって処理を分ける方法（if 文）
4 回目	条件によって処理を分ける方法（if 文） 変数を一括管理する方法（配列）と繰り返し処理（for 文）	変数を一括管理する方法（配列）と繰り返し処理（for 文） 配列の便利な使い方
5 回目	配列の便利な使い方 まとめ	まとめ

作」に関する総合的理解を促すことを目的としている。このような科目構成から専門科目としてプログラミングのみを学ぶ授業に比べ、限られた時間内で効果的な指導を行うことが必要であった。

3.3.2　授業デザインと授業形態

対象とした科目の中でプログラミングを学ぶ機会は、先述のように 1 授業 90 分で計 5 回しかなく、学習する時間が限られていることから、学習内容は、プログラミングの基本的な要素である変数と制御構造を中心とし、関数や構造体は取り上げなかった（表 2 参照）。本授業では、教材の設計意図を踏まえ、講義と写経型学習教材による個別学習を取り入れた授業形態を採用し、講義では短時間にポイントを絞って解説を行った。また、受講生は、自立的に学習できるよう開発した写経型学習教材に従って、学生間で学習の進度を合わせることなく、学生各人の進度に応じて学習を進めた。受講生がつまずいた場合は、教員やティーチングアシスタント（1 人）がサポートに入り、直接対応した。

なお、演習には、京都大学の教育用コンピュータ端末を用い、C 言語の処理系には、インストールが比較的容易で学生が自宅で作業する環境の構築が容易であることや日本語でのエラー表示機能があることなどから MinGW（Windows 版）を、プログラムの編集には汎用のテキストエディタである TeraPad を用いた。

3.3.3　受講生の構成

本実践での受講生は 15 人で、その学部別の内訳は、総合人間学部（6）、法学部（3）、経済学部（3）、文学部（2）、理学部（1）であった。この授業は、プログラミング初学者を想定したものであるが、経験者の履修も可能であり、本実践では、JAVA など他のプログラミング言語の既習者が 2 人であった。

3.3.4　検証方法

開発したテキスト教材を検証するため、受講生から教員およびティーチングアシスタントへの質問とその際の参与観察の結果における「つまずき」どころとその件数、内容について調べた。さらに、受講生のプログラミングに関する理解度を測るため、各授業終了時に「確認テスト」を、また、第 5 回目の授業の最後に「最終テスト」を筆記式で行った。確認テストでは、受講生の進捗状況に応じて、順次処理、条件分岐処理（if 文）、繰り返し処理（for 文）に関する選択式（3 択）の問題を 3 問、プログラムの作成問題を 1 ないし 2 問をそれぞれ課題とした。また、第 5 回目の授業で実施した最終テストでは、1）文法エラーの間違い探し、2）変数・画面への出力（printf 文）・キーボードからの入力（scanf 文）に関するプログラムの作成問題、3）if 文に関するプログラムの作成問題、4）for 文と配列に関するプログラムの作成問題をそれぞれ課題とした。

3.4　つまずき多発箇所と学習への影響

本授業では、全授業回を通して、受講者 15 人のうち既習者 2 人を除く初学者 13 人全員から延べ 63 件の質問が見られた。このうち 44 件は、「コンパイルエラー表示の意味がわからない」、「エラーの原因がわからない」など、自ら対処できないために教員に質問をしたものであり、本稿では、これらをつまずきどころとして取り扱った。これらのつまずきについて、「コンパイル手順」「エラー対応」「基本操作」の 3 つに類型化したものを表 3 に示す。この類型

のなかで「コンパイル手順」および「エラーの対応」と実行までの手続きに係るつまずきが計38件と、とりわけ多くみられた。こうした「つまずき」が多発したことなどの理由により受講生のなかには、サンプルプログラムを1回ずつ試すだけで自ら一部改変して試すことができなかった者が見られたほか、少なくとも3人はサンプルプログラムを入力しただけで、実際に動作させるまでに至らなかった。

4. 2011年度の実践

4.1 「つまずき」の要因に関する分析とテキスト教材の改良

2009年度の実践では、「繰り返し処理のfor文」の理解を計る確認テストおよび最終テストにおいて、それぞれ、7人および11人と初学者の過半数が誤答している（表4参照）。なお、ここでは、各問題の回答に1問でも誤りがあった者を「誤答者」としている。この結果は、開発した教材が、当該概念を理解するために十分でなかったことを示していることから、改善の必要があるものと考えられる。また、表3で示すように、「コンパイル手順」や「エラー対応」といった実行までの手順に係るつまずきについては特に注視して関係箇所を見直す必要があるという結果が得られた。「繰り返し処理のfor文」の両テスト結果において過半数の誤答が見られたことの原因が「実行までの手順に係るつまずき」を軽減するための工夫が足りないことであるという結論を導くことはできないが、少なくとも「写経型学習」を行う上で、実行に係る作業を滞りなく最後まで行うことは最低限必要な作業であり、学習のための必要条件を欠いていたともいえる。これらの必要条件を満たしたうえで、さらにテスト結果に良い変化が見られないのであれば、「実行までの手順に係るつまずき」以外の要素、ここでは、「作業を介した理解」の過程にも改善の要素があると考えねばならない。このため著者らはまず、「実行までの手順に係るつまずき」を軽減させるための改善を進めることとした。

4.1.1 「実行までの手順に係るつまずき」を軽減させるための改善

2009年度の実践では、「手順通りやってもコンパイルできないがどうしたらよいか」、「コンパイルエラー表示の見方（意味）がわからない」といった質問内容が多く見られたほか、

(1) コンパイルの方法が分からないと述べる受講生の一部は実際にはテキスト教材を参照せずに作業を進めていた。

表3　2009年度および2011年度授業実践におけるつまずき件数と事例

つまずきの類型	2009年度		2011年度	
	つまずき件数	つまずき事例	つまずき件数	つまずき事例
コンパイル手順	12	コンパイルしたが前回のプログラムが実行された（コンパイルせずに実行コマンドのみ入力していた）など	1	手順書に従って進めたがコンパイルできなかった（原因不明）など
エラー対応	26	エラー表示が出たが意味が分からないため対処できなかったなど	3	不正な文字の使用を意味するエラー表示が出た際，エラー表示の意味は分かったが，エラー箇所が見つけられなかったなど
基本操作	6	・CapsLockの解除方法がわからない ・全角と半角の切り替え方がわからないなど	3	・NumLockの解除方法がわからない ・コマンドプロンプトのウィンドウを閉じるべきかそのままで実行すべきかわからないなど

表4　2009年度および2011年度における確認テストと最終テストの誤答者数

設問の内容		2009年度の誤答者数	2011年度の誤答者数
順次処理	確認テスト	0	0
	最終テスト	0	0
条件分岐処理のif文	確認テスト	0	0
	最終テスト	3	2
繰り返し処理のfor文	確認テスト	7	0
	最終テスト	11	2

(2) テキスト教材を参照しながらコンパイル作業を行っていた受講生の中にも手順を進める上で混乱が生じていた。なかでも、コンパイルが毎回必要な作業であると認識しておらず、2回目以降のプログラムではコマンドのみ入力している受講生がいたなど、作業の開始点に関して混乱が見られた。

(3) 本文中で、いくつかの典型的なエラーの原因となる「間違いやすい文字」、「見えない文字」、「カッコのとじ忘れ」などについて注意を促す記載があり、これらに加え別紙でエラーメッセージの実例とその解説も2例紹介していたが、これだけでは実際のエラーに対処できていなかった。

などの観察結果が得られていた。そこで、以下の3つの方針に従ってテキスト教材の改善を行い、一連の手順とエラーへの対応についてまとめた「手順書」を別冊子として作成した。

(1) コンパイルの手順を学習の初期段階にすでに理解したはずの項目としてではなく、常に傍らにおいて参照すべきものとして認識させるため、手順書を別冊子にして受講生に配布する。

(2) コンパイルの手順を、「C言語を学習する際に一度だけ行う作業」、「毎回の授業開始時などC言語を学習するごとに行う作業」、「C言語のプログラムを作成・修正するごとに行う作業」として、作業の開始点を明確に示すとともに、各手順についても詳細に示す。

(3) エラーメッセージとエラーの実例を、発生頻度が高いと思われる「"（ダブルコーテーション）が足りない」、「スペリングの間違い」の2例を加え、計4例に増やす。

4.2 改善した教材の実践における検証

2009年度の授業では、コンパイルやエラー発見におけるつまずきに関しては提示した手順が分かりにくかったことや、一部の学生が手順を記したページを参照していなかったことから、さらに詳細な手順を別冊で用意して配布することで改善が可能であると考え、2011年度の後期に京都大学において実施された同名の授業において、改善したテキスト教材を用いた実践と検証を行った。

4.2.1 対象とした授業と受講生の構成

本研究の対象とした授業は、2009年度と同様に京都大学の全学共通教育科目「情報科学演習」であり、2011年度の後期の授業を対象とした。受講生は18人で、学部別の内訳は、経済学部（6）、工学部（4）、農学部（3）、総合人間学部（2）、理学部（2）、文学部（1）であった。また、プログラミングの既習者は4人であった。なお、2011年度の授業内容、「確認テスト」および「最終テスト」は、2009年度と同様のものを使用した。

4.2.2 2011年度の授業実践におけるつまずき箇所と改善状況

2011年度の授業では、全授業回を通して、受講生の9人から延べ14件のつまずきが確認されたが、とりわけ、2009年度につまずき箇所として多く見られた「コンパイル手順」や「エラー対応」といった「実行までの手続きに係るつまずき」については、受講生数（初学者数）が、2009年度の15（13）人から、2011年度の18（14）人と若干ながら増加しているにもかかわらず、表3で示す通り、つまずき箇所の件数はわずか4件と大幅に減少した。

4.2.3 テストの結果

2009年度のテストでは、「繰り返し処理のfor文」に最も多くの誤答が見られ、確認テストで7人、最終テストで11人の誤答者があった。これに対して、2011年度では、最終テストの際、「条件分岐処理のif文」と「繰り返し処理のfor文」において各2人の誤答者が見られたのみであった。テスト結果の詳細を表4に示す。このように、プログラムを構成する基本的な概念の理解を測る問題、とりわけ「繰り返し処理のfor文」において誤答者数の著しい減少が見られた。

5. 考察

5.1 自立的な作業遂行のための支援

本研究では、プログラミングの学習において学習者の自立的な作業遂行と作業を介した理解に配慮した「写経型学習教材」を開発し、授業での運用を試みた。こうして開発した教材を用いたことに関し、2009年度の授業実践では、開発したテキスト教材に「コンパイル手順」を示していたものの、これに係る「つまずき」が多数見られた。一方、2011年度の実践では、コンパイルの手順やエラー発生時の対応など、実行までの手続きに係る「手順書」を作成し、

別冊子で配布した結果、つまずき件数については2009年度の38件から大幅に減少し、4件となった。このような学習者に対する提示方法の工夫については、Brittonら（1993）も教材の書き換え（記載順序の修正）や追記（見出しや下線の追記など）だけでも、教材の種類に関係なくおおむね学習の向上が認められたと報告しており、わずかな変更であっても適切な修正あるいは改善であれば得られる効果は少なくないものと思われる。

5.2 「作業の自立的遂行」の改善による概念理解への影響

2011年度の実践で使用した「手順書」は、直接的には、作業の自立的遂行の「つまずき」の軽減を目的に改善されており、同年度の実践では、意図したとおり、実行までの一連の過程におけるつまずきは大幅に減少している。加えて、この改善の効果は、作業におけるつまずきの減少にとどまらず、概念の理解を測るテストにおいても誤答者数の減少として顕在化しており、特に「繰り返しのfor文」のテスト結果において顕著である。

こうした結果は、「写経型学習」過程がfor文の概念の理解に一定の効果を持つことを示唆しており、一部限定的ながらも、「写経型学習」過程の必要性と具体的効果をプログラミング学習分野において初めて量的に示した事例となった。

阿部（2009）が「for文の入れ子的な利用も初学者が典型的に困難さを抱えるものである」と指摘しているように、for文の概念を理解させるためには相応の工夫が必要とされるが、本事例で用いたテキスト教材ではfor文ドリルといった形で「写経型学習」過程の徹底を図っており、2011年度の実践の際、テキスト本文と同ドリルに掲載されたプログラムの実行過程が滞りなく進んだことによってif文と同程度の正答者数を示すまでに至ったと考えられる。

河内谷（2006）が指摘しているように、「代入という概念がすでに数学で学習済み」であるのに対して、for文については「類似の概念を目にしたことがない」のだとすれば、学生は、類似性の高い既知の概念を見つけるのではなく、新たに概念を習得しなければならない。本実践では、異なる複数の事例から共通の要素を抽出して新たな概念を形成する、いわゆる帰納の過程によって、それを実現していたと考えられる。

5.3 本研究で用いた概念理解のための工夫とその効果

本研究で開発したテキスト教材では、当該概念を学ぶにあたり「何を目的として何をしているか」について予めテキスト教材の中で明確に説明しており、こうした方略により「作業を介した理解」の促進を期待している。この改善点が効果的に作用したか否かについては、授業実践の観察結果およびテスト結果などによって直接的に示されるデータはないが、少なくとも、授業中に見られた質問の中に「何を目的として何をしているかがわからない」といった類の質問は見られなかった。また、テスト結果を見ても2011年では、if文とfor文の最終テストにおいてそれぞれわずか2例の誤答が見られたのみであることを考慮すれば、写経型学習過程にそうした工夫を加えるという簡便な方略だけでも、多くの学生が当該概念を理解するに至ったと解釈できる。一方、わずか2例であっても、理解できなかった学生も見られた。そのような学生には、ここで用いた「何を目的として何をしているか」について予めテキスト教材の中で明確に説明するといった方略だけでは十分でなかった、という可能性も否定できない。

6. おわりに

本研究では、プログラミング初学者がサンプルプログラムにある命令、文法などを模倣するところから一部改変したり、組み合わせたりするなどして試していくまでの過程を「写経型学習」と規定し、まずこの「写経型学習」を中心とした学習過程で問題点を調査した。その結果、

(1) 写経型学習を遂行する上で自立的に作業することができない
(2) 写経型学習の過程から目的とする内容を学び取ることができない

といった「作業遂行上のつまずき」と「作業を介した理解に係るつまずき」の2つに類型化することができた。

この結果を踏まえ、著者らは、学習者の自立的な作業遂行のために「コンパイルの手順やエラー発見の方法についてその手順を掲載すること」と、作業を介した理解に配慮するために「学習の目的、学習の仕方など、学ぶ内容とともに学び方を明示すること」という2つの方略によって問題解決を図ることを企図して、「写経型学習教材」を開発した。

そして、2009 年度の後期に京都大学で開講された科目「情報科学演習」において開発したテキスト教材の検証を試みたところ、コンパイルの手順やエラー発生時の対応などにおける「つまずき」が多発したため、2011 年度の実践において、コンパイルとエラー対応に係る記載に改善を加えて別冊子でこれを配布した結果、手順に係るつまずきが大幅に減少した。また、作業上のつまずきの減少に合わせて、とくに for 文の理解を問うテストにおいて誤答者数の大幅な減少が見られた。2009 年度の実践における作業上のつまずきの多発については意図したものではないが、その改善を試みるなかで、「写経型学習」過程が滞りなく進むことで for 文の理解が向上したことが示された。このことは、「写経型学習」それ自体が for 文の理解に一定の効果を持つことを示唆している。

なお、「写経型学習」の過程から目的とする内容を学び取ることができないといった問題については、本実践で用いた改善が効果的に作用したか否かを直接的に示すデータは得られなかったが、「何を目的として何をしているかがわからない」といった類の質問は見られなかった上、テスト結果を見ても 2011 年度では、if 文と for 文の最終テストにおいてそれぞれわずか 2 例の誤答が見られたのみであった。このことを考慮すれば、写経型学習過程にそうした工夫を加えるという簡便な方略だけでも多くの学生が当該概念を理解するに至ったと解釈できる。

このように、本稿で示した事例では、「写経型学習」に係る改善を目的とした教材を開発していく中で、学習の目的を明示し、実行に至るまでの作業が滞りなく進むよう丁寧な支援を行うことで、従来つまずきが多発されるとされた for 文の理解に関して、8 割以上の学生を理解に導くことができた。こうした結果は、学習課題や対象となる学習者層について限定的であり、さらに、実験レベルのデータではなくあくまで実践の中で得られた結果ではあるものの、「写経型学習」過程の重要性とその改善に係る具体的方略をプログラミング学習分野において初めて量的データを伴って示したものである。

7. 今後の課題

授業の際、プログラム実行までの過程については、ステップごとに操作方法を指示しながら、全員同時に作業させるなど、一斉授業の形態でも実施できるが、学習者に複数の事例を体験させたり、また、サンプルプログラムの一部を改変したりするなどの試行錯誤の過程まで視野に入れると、やはり、テキスト教材は、Personalized Learning 環境を想定して開発すべきであると著者らは考える。

こうしたことを踏まえ、さらに、直感的に理解しやすい教材に改善できればなおよい。著者らは、本研究と並行して、直感的に理解することを目標とした教材の作成を進めており、「『視覚的顕在化』に着目したプログラミング学習教材の開発と評価」（岡本ら、2013）としてまとめたが、この教材は、特殊なプログラミング環境を用いていることや特殊な装置を新たに用意しなければならないといった点で、一般的な授業で用いるには負担が大きい。今後は、こうした研究から得られた知見を本実践で用いた汎用性の高い教材に反映させ、広範な場での活用を想定した「写経型学習」教材の開発を検討していきたい。

引用文献

阿部圭一（2009）「C 言語によるプログラミング教育についての省察」『情報処理学会研究報告』コンピュータと教育研究会報告，第 15 号，205–212 頁.

Britton, B. K., Gulogoz, S., & Glynn, S. (1993) “Impact of good and poor writing on learners: Research and theory”, In B. K. Britton, A. Woodward & M. Binkley (Eds.), *Learning from textbooks: Theory and Practice* (pp. 1–46). New Jersey: Lawrence Erlbaum Associates.

江木鶴子，竹内章（2009）「プログラミング初心者にトレースを指導するデバッグ支援システムの開発と評価」『日本教育工学会論文誌』，第 32 巻，第 4 号，369–381 頁.

深町修一（2010）「文系の学生に対するコンピュータプログラミング教育の一考察」『福岡国際大学紀要』，第 23 巻，39–45 頁.

長谷川聡，山住富也（1997）「プログラミング教育と学習者のイメージ形成」『名古屋文理短期大学紀要』，第 22 巻，9–14 頁.

長谷川聡，山住富也（1998）「プログラミング教育と学習者のイメージ形成（その 2）」『名古屋文理短期大学紀要』，第 23 巻，9–14 頁.

Kaminski Jennifer A., Vladimir M. Sloutsky, & Andrew F. Heckler (2008). *The Advantage of Abstract Examples in Learning Math. Science*, 320, pp. 454–455.

河内谷幸子（2006）「プログラミング言語の学習法―for 文の理解に関する認知心理学的分析―」『言語と文化』，第 3 号，19–35 頁.

河西朝雄（2008）『なぞりがき C 言語学習ドリル―書き込んでいくことでしらずにしっかりマスター―』，技術評論社.

日本教育工学会編（2000）『教育工学事典』実教出版.

岡本雅子，村上正行，喜多一，吉川直人（2010）「初学者を対象とした自習中心のプログラミング教育の教材開発と評価」『情報処理学会研究報告』情報教育シンポジム SSS2010 論文集，87–94 頁.

岡本雅子，村上正行，吉川直人，喜多一（2013）「『視覚的顕在化』に着目したプログラミング学習教材の開発と評価」『日本教育工学会論文誌』第 37 巻，第 1 号，35–45 頁.

太田信宏（2009）「Java プログラミング教育に関する一考察」『立教大学女子短期大学部研究紀要』第 52 巻，1–16 頁.

柴田望洋（2004）『新版 明解 C 言語入門編』，ソフトバンククリエイティブ.

高橋麻奈（2012）『やさしい C　第 4 版』，ソフトバンククリエイティブ.