プログラミング導入演習に対して 学習支援システムと自己調整学習が与える影響の分析

An Analysis of Effects on Introducing a Learning Support System and Self-regulated Learning in Fundamental Programming Exercises

國宗永佳 Hisayoshi Kunimune† 仲林 清 Kiyoshi Nakabayashi†

Summary

筆者はプログラミングの導入授業を、与えられた問題を決められた操作や制御構造のみの組合せによって解決する思考であるアルゴリズム的思考の教育と位置付けて、2008年度から実践している。この実践において、当初は制約を加えた日本語によるアルゴリズムの記述を行っていた。しかし、個別指導に割く時間の減少などから学習効果が低減したため、ビジュアルプログラミング環境と授業運営支援機能を有する学習支援システム AT を開発し導入した。年度ごとの授業改善とシステムの改善によって、システム導入後に学習効果が向上したことが分かった。更に新たな取組みとして、他の授業で自己調整学習に関して学ぶことが、プログラミングの導入授業の学習活動に及ぼす影響についての調査を行った。本論文では、これまでの実践の概要と評価、学習支援システム AT の概要、自己調整学習に関する学びによる学習活動への影響調査について述べる。

Key Words

アルゴリズム的思考、プログラミング教育、ビジュアルプログラミング環境、自己調整学習

1 はじめに

本論文の第一著者は2008年度から情報工学を専攻する大学1年生に向けたプログラミングの導入授業・演習を、「アルゴリズム的思考」の教育と位置付けて実施している⁽¹⁾。ここではアルゴリズム的思考を「与えられた問題を、決められた操作や制御構造のみを組み合わせて解決する思考」と定義している⁽²⁾。この思考はプログラムを作成するために重要なものであり、プログラミング言語には依存しない。2006年にWingが提唱したComputational Thinking⁽³⁾(和訳「計算論的思考」⁽⁴⁾も存在する)においては、シミュレーションなどの近似解の利用や不確定な状況に対するヒューリスティックな推論など、より高度な思考についても言及があるが、基礎的な部分については共通した考え方であると考えている。

上記の実践において、当初は語彙を統制した日本語を用いた擬似コードを使用して問題の分解・整理を行っていたが、後にビジュアルプログラミング環境と授業運営機能を備えた学習支援システム AT を開発して導入した。また、新たな取組みとして自己

調整学習がプログラミング演習に与える影響についての調査を行った。

本論文では、**2.**で本邦におけるアルゴリズム的思考教育の状況について概観する。**3.**では過去に行った実践とその評価について述べる。**4.**では学習支援システム AT の概要について述べる。**5.**では自己調整学習がプログラミング演習に与える影響について述べる。最後に**6.**で本論文をまとめる。なお、本論文は研究会等で既発表の内容^{(1),(2),(5)}を基に、大幅に加筆・再構成したものである。

2 本邦におけるアルゴリズム的思考教育の状況

本邦におけるアルゴリズム的思考に関する研究は、1960年代に端を発して以来^{(6),(7),(8)}、数多く行われている⁽⁹⁾⁻⁽¹⁶⁾。文献⁽¹⁷⁾に詳述しているが、当初は幾何算数の教授に際して論理操作を表現する手段としてアルゴリズムモデルを適用するなど、教授行為への応用から始まっている。1990年代以降には、アルゴリズム教育の提案や実施例が数多く示されている。ここでは、これらの事例を操作の体験・一般化・記述という三つの行為に分類する。

操作の体験とは、問題解決のための手続きの概念化を指す。ここに分類される多くの事例では迷路探索⁽⁹⁾やハノイの塔⁽¹⁰⁾、探索・並べ替え^{(11),(12)}、組合せ最適化問題⁽¹³⁾などの、アルゴリズムによって解

† 千葉工業大学, 習志野市
Chiba Institute of Technology, Narashino-shi,
275-0016 Japan

決することに適した課題が用いられている。

操作の一般化とは、操作体験を帰納的に汎化させることであり、特定の問題を解決した手法が、類似の課題や一般的な問題の解決に適用できることを発見させる事例や誘導している事例^{(11),(12)}が多く見られる。

以上の操作の体験と一般化のプロセスは、アルゴリズム的思考に関する概念を理解している段階であると考えられる。一方、次の段階である操作の記述では、概念化された手続きを特定の形式で定式化することが求められる。この段階では、表現上の制約を設けた言語（制約付き日本語⁽¹³⁾、Squeak⁽¹⁴⁾、独自に設計した言語⁽¹⁵⁾など）や、アルゴリズムを記述するために標準的に用いられる図（フローチャート、PAD⁽¹⁶⁾など）が多く用いられてきた。これらの記述にはPC上の開発環境が用いられることも多い。

本論文で述べる実践においても、操作の体験・一般化・記述のプロセスについて、特に体験と記述に重心を置いて教授している。操作の体験においては、当初は並べ替えを題材としていたが、後に学習者が想起しやすい状況を題材とした問題設定を行った。

3 授業実践と評価

従来プログラミングの導入教育を行う授業において、プログラミング言語の文法や、変数・配列などのデータ構造や制御構造などの基礎的な概念を教える中で、問題解決のための分解・整理のための考え方を経験的に養うという教育を行っていた。その結果、アルゴリズムを十分に検討せずに、作成したプログラムの動作から場当りの問題点を修正する学生が顕在化してきた。このような方法で正しく動作するプログラムを作成することは困難であり、プログラムを作成する前にアルゴリズムについて詳細な検討を行う必要がある。

そこで、プログラミング言語教育を行う前段階の授業に、プログラミング言語を用いないアルゴリズム的思考教育を導入した。本章では、アルゴリズムの記述に日本語による擬似プログラムを用いた2008～2009年度の実践と、学習支援システムを導入した2010～2012年度の実践について述べる。

3.1 2008～2009年度の実践

2008～2009年度の実践では、並べ替えを題材としてアルゴリズムの学習を行った⁽¹⁷⁾。始めに学生自身に数値を記入したカードの並べ替えを体験させ

た上で、それをアルゴリズムとして書き下すことの難しさを伝える。この段階では、アルゴリズムを記述する際に用いることのできる操作や制約などについて統制しておらず、学生ごとに異なる粒度の操作を用いたアルゴリズムを考案している。その後、並べ替えアルゴリズムの一部分となる最大値の選択や値の入替えなどを記述するために必要となる変数や配列、条件分岐、繰返しなどの概念や操作を順次導入した。また、各回の授業で新たに導入した概念や操作については、基本的な練習問題を提示し、学習者が作成したアルゴリズムを教員が個別に添削指導した。

以上のような手順で、最終的には簡単な並べ替えアルゴリズムを実現するために必要な概念や操作とその記述方法を導入した。プログラミング言語を学習する前の学生に導入するアルゴリズム記述方法として、記述上の制約を設けた日本語を用いることにした。その結果、例えば「なでしこ」⁽¹⁸⁾などの日本語で記述可能なプログラミング言語に近い記述を行うことになったが、先に述べた場当りの実行ができないよう、アルゴリズムを紙上に手書きすることになった。

2008年度の授業実践では、上述のとおり並べ替えを題材としていたため、二つの数値の比較による最大値や最小値の探索と、値の入替えを主な話題として扱っていた。2008年度の授業実践後、授業では扱わなかった、配列の操作を含まず数値の計算を主体とした課題を提示した。これは割り算や余り算を用いずに、与えられた数値が3の倍数かを判定する課題（3.3に詳述）である。その結果、授業で扱った課題と授業実践後に提示した課題で共通した操作や概念が定着したことが分かった。一方、繰返しの使い方については、授業内と授業実践後の課題で次のような差異がある。前者では、数値が記入されたカード（配列）の先頭から末尾までたどるように添え字を変化させる。後者では、与えられた数値から一定の値（3）を下回るまで一定の値（3）を引いていく。このような差異があった点については、十分理解できていないことが明らかになった。

そこで、2009年度には、2008年度の授業実践で題材とした数値の比較や入替えに加えて、数値の計算を主体とした課題（例えば、事前に昇順に並べられた重複を含むデータに対して順位を付与する課題）も取り上げた。しかし、2008年度と2009年度で共通して授業実践後に提示した課題（割り算や余り算を用いずに、ある数値が3の倍数かを判定する）の解答を比較した結果、改善が見られなかった⁽¹⁹⁾。

3.2 2010～2012 年度の実践

2010 年度から、実行順序や変数の値の変化を強く意識付け、アルゴリズムの動作に対する理解を深めるために、各手順における変数の値の変化をトレースする活動を導入した。また、学習支援システム AT を開発し導入した⁽²⁰⁾。AT の概要については 4. で述べる。

AT は、ビジュアルプログラミング環境でプログラムを作成・実行することができ、作成したプログラムを電子的に提出することが可能である。また、教授者も提出されたプログラムを実行することが可能であるため、従来は紙に書かれたアルゴリズムの動作を確認しながら行っていた採点の負荷が軽減された。

2009 年度の授業と比較すると、取り上げた題材や導入の順序については同様であったが、上述のトレース活動を導入した点と、AT を用いたことで採点の負荷が下がり、個別指導の回数が増えた点が異なる。また、ビジュアルプログラミングでは操作のブロックの選択・配置と変数名や数値の記述のみでプログラムを作成できるため、日本語による記述の制約についての説明が不要になった。この結果、概念や操作の説明により多くの時間を割くことが可能になった。

2010 年度の授業では、3.1 末尾で述べた問題点を踏まえて改善を行った。2009 年度と 2010 年度の学生について、同一の課題の解答を比較した結果、一定の改善が見られた⁽²⁰⁾が、2008 年度の学生との比較では改善が見られなかった。

そこで 2011 年度の実践には、以下のように使用する要素や、取り上げる題材、課題の形式について大幅な変更を行った⁽²¹⁾。使用する要素については、繰返しの種類とデータ構造の絞り込みを行った。従来、繰返しとしては for 型の繰返し（a から b まで 1 ずつ増加しながら繰り返す）と、while 型の繰返し（条件が成立する間繰り返す）の 2 種類を使用していた。しかし、両者はほぼ同等であるため、while 型の繰返しのみを扱うことにした。また、データ構造については変数と配列を導入していたが、題材の選び方によっては配列の概念は必須ではないと考え、変数のみを導入した。

配列を導入しないため、従来取り上げていた並べ替えを題材とすることが困難になった。そこで、データ構造として変数のみを用いて実現可能な題材を複数用意した。また、問題自体の理解を容易にするために、日常生活（お釣りの硬貨の枚数、バスの運賃計算など）や高校までの学習内容に現れる事柄

（輸血可能な血液型の組合せなど）を題材として多く取り上げた。

従来の実践においては、提示した問題を解くアルゴリズムを一から作成させる形式の課題を提示していたが、アルゴリズムを作成することが困難な学生も存在した。そこで、課題を読み動作を答える形式や、アルゴリズムの一部を穴埋めする形式など、解答形式の異なる課題も提示し、ある問題を解くためのアルゴリズムをどのように構成するかを知る機会を増やした。これらの課題の難易度や適切性については、学習者や課題の分析に広く用いられる S-P 表⁽²²⁾などを用いて評価を行い⁽²³⁾、年度ごとに改善を重ねている。

また、従来はその日の授業で取り扱った操作に焦点を当てた課題を提示していたため、用いる操作を推測しやすい状況であり、問題に応じて使用する操作を適切に選ぶという過程が重視されていなかった。そこで、課題演習と解答の解説を主とした授業を設け、課題を解く機会と、必要な操作の選択について考える機会を増やした。

2012 年度の実践では、AT にステップ実行機能を追加した。このことによって、従来実施していたトレース活動と同様に、システム上で作成したアルゴリズムの動作を順に追うことが可能になった⁽²⁴⁾。また、各自で動作の確認を十分に行うことができるようになったため、従来は板書や資料提示によって行っていた動作の提示を、各自で確認するように変更した。このことによって、各操作の導入に必要な時間が短縮され、課題演習により多くの時間を割くことが可能となった。

3.3 評価

上述した実践について、評価を行った。アルゴリズム的思考の授業は毎年度、学部 1 年生の前期に実施した。授業の実施後、夏休みを挟み、後期に実施している授業の第 1 回目において、毎年度同一の課題を提示した。

課題は「30 以下のある整数 N が、3 の倍数であるかを判定し、『3 の倍数である』または『3 の倍数ではない』という結果を表示するアルゴリズムを、除算と剰余を用いずに作成する」というもので、解答時間は 30 分間としている。2008 年度から 2011 年度までは解答を紙に書いていたのに対して、2012 年度には AT で解答を記述した。紙面での解答と同様の状況にするために、AT ではプログラムを実行できない状態にした。

この課題の解答に対して、下記の五つの評価項目

を設定し、それぞれを満たしていれば1点、満たしていなければ0点とし、5点満点で採点を行った。

- (1) 大枠の構成
繰返しの中で何らかの判定を行っている
- (2) 3の倍数の判定
3ずつ足すか引きながら判定している
- (3) 両端の判定
Nの全ての場合について判定している
- (4) 単純なミス
文法的な誤りなどを含まない
- (5) 論理的なミス
変数の初期化忘れ、結果の複数回表示などを含まない

ATの導入前(2008～2009年度)と導入後(2010～2012年度)に分け、各項目について帰無仮説を「導入前後の平均値に差はない」、有意水準を1%としてWelchの検定を行った。この結果、(3)以外の項目において導入後に有意な改善があることが分かった。各項目における合計点の平均値と標準偏差(括弧内)、p値を表1に示す。

導入前後においてはATの有無だけでなく、課題形式の多様化や、授業展開の改善、課題数の増加など様々な授業改善も並行して行っており、各項目の平均値が向上したことについて、一概にATを導入したことによる効果ということとはできない。しかし、ATを導入することによって、学習者が個別に

表1 ATの導入前後における各評価項目平均点
(括弧内は標準偏差)

項目	導入前 N=187	導入後 N=271	p 値
(1)	0.82 (0.38)	0.95 (0.22)	7.47e-5**
(2)	0.78 (0.41)	0.91 (0.29)	3.44e-4**
(3)	0.80 (0.40)	0.82 (0.39)	0.649
(4)	0.71 (0.45)	0.88 (0.32)	1.35e-5**
(5)	0.42 (0.50)	0.64 (0.48)	3.26e-6**
合計	3.50 (1.66)	4.21 (1.22)	1.16e-6**

(** : $p < 0.01$)

基礎的な概念の確認を行うことが可能になり、教員の課題採点の手間も軽減されるなどしたため、課題の添削などの個別指導や課題演習に割く時間が増加した。このことが特に評価項目(1)、(2)、(5)の理解向上に寄与したと考えている。

評価項目(4)については、導入後の平均値が高く、中でも2012年度の平均値が高い(2008年度から順に、0.74, 0.68, 0.79, 0.89, 0.95)。導入前と2010年度は書き方に制約を加えた日本語によって解答を紙上に記述していたのに対して、2011年度はビジュアルプログラミングのブロックを簡易化したものを紙上に記述した。また、2012年度はAT上で記述したため、文法的な誤りがほぼなくなったことが原因である。

一方、評価項目(3)については個別指導の量や課題数による影響が見られず、効果的な学習が行われていない可能性がある。また、今回評価を行った問題と異なるタイプの問題に対する解決能力についても確認し、学習効果を測定する必要がある。

4 学習支援システム AT

3.1で述べた2008～2009年度の実践ではアルゴリズムを日本語で記述していたが、計算機で実行できないため提出物の評価に掛かる負担が大きく、個別指導に十分な時間を割くことができないという問題があった。この問題を解決するためには、計算機で実行可能なプログラミング言語による記述に変更する必要があったが、以下の2点の問題が考えられた。

- ・アルゴリズムの誤りに対する場当りの対応が可能になること
- ・文法誤りなどへの対応が必要になること

前者の問題については、教員がプログラム実行の可否を制御することで対応することにした。そのため、教員が試行錯誤を行うべきと考える場合には制限なくプログラムを実行することが可能である。

後者の問題については、岡本ら⁽²⁵⁾が「写経型学習」における初学者のつまずきのうち、「非本質的な認知的負荷を伴う学習過程によるつまずき」として位置付けている。ここでは以下のような事例が挙げられており、写経型学習に限らずプログラミング学習の初期に共通した問題であると考えている。このような学習のつまずきを避けるために、ビジュアルプログラミングに着目した。

“「expected ‘;’ before ‘return’」という「return」

図1 ATのビジュアルプログラミングエディタ

という命令の前に文法エラー (syntax error) があるという表示から該当箇所を発見することができない” (文献 (25) から引用)

4.1 ビジュアルプログラミングの特徴

ビジュアルプログラミングとは、図形の組合せによってプログラム作成を実現することである。例えば、プログラムの処理の流れを記述するフローチャートから実際に動作するプログラムを作成することのできる PenFlowchart⁽²⁶⁾や、マサチューセッツ工科大学において開発され、近年本邦においても小学生など初学者を対象としたプログラミング教室などで多く使用されている Scratch^{(27),(28)} のようなビジュアルプログラミング環境が多数開発されている。

これらの特徴は、上述したように図形の組合せによってプログラムを作成できることにある。それぞれの図形は、形式言語によるプログラミングにおいて入力されるキーワードなどを表現したものである。そのため、ビジュアルプログラミング環境においては、プログラム作成者が自ら決める必要のある名前や値など、必要最小限の文字を入力するだけでプログラムを作成することができる。この特徴により、文法誤りを起こさずにプログラムを作成することができる。

4.2 ATの概要

筆者が開発している AT は、上述した特徴を備えたビジュアルプログラミング環境を提供する。ほかのビジュアルプログラミング環境にない特徴とし

て、以下のような授業運営支援機能を備えている。

- 学習者によるプログラム動作の確認を支援する
- 教員が作成する課題と、学習者が作成するプログラムを体系的に管理し、教員・学習者間で共有する
- 教員が各学習者のプログラムに対して直接評価・指摘を与えられる
- 教員による成果物の評価を支援する

また、AT は Web アプリケーションとして実装されているため、PC やタブレット端末など様々な環境において利用可能である。

AT は主に以下の機能を有する。

- ビジュアルプログラミングエディタ (図1)
図形 (AT ではブロックと呼んでいる) によるプログラムの作成・実行・提出
Google 社による Blockly⁽²⁹⁾ を基に開発
- 学習者用画面
教員が提示した課題の内容や採点結果を確認
- 教員用画面
学習者に提示する課題の編集や採点の支援 (図2)
- 管理者用画面
システムに関わる情報 (ユーザ情報など) の管理

学習者は図1左側のメニューから必要なブロッ

秀	これでOKです.	成功	成功	成功	成功	Success
秀	これでOKです.	成功	成功	成功	成功	Success
可	このアルゴリズムでは...	失敗	成功	失敗	失敗	Failure
可	このアルゴリズムでは...	警告	警告	警告	警告	Failure
良	100円玉と10円玉...	失敗	失敗	成功	失敗	Failure
可	このアルゴリズムでは...	成功	失敗	失敗	失敗	Failure
可	このアルゴリズムでは正...	失敗	失敗	失敗	失敗	Failure

図2 ATの教員用画面の一例

各学生が提出したプログラムが各行、教員が作成したテストケースに基づくプログラムの採点結果が右側のボタンの列に対応する（緑＝正常、黄＝異常、赤＝実行時エラー）を示す。

クを選択し、図1中央部でブロックを接続することによってプログラムを作成する。本実践ではATを用いた授業を行った後に実施されるC言語によるプログラミング科目に接続する必要があるため、基本的にはC言語のプログラムと直接対応するようなブロックを用意している。ただし、例えば「もし～ならば」、「～の間、繰り返す」のように、表現についてはなるべく平易なものを用いている。

またプログラム実行時に、事前に設定した任意のブロックで一時停止できる機能や、変数の値の変化とプログラムによる出力を逐次表示する機能（図1右側）によって、学習者がプログラムの動作を確認することを支援している。

学習者が提出した解答（プログラム）について、教員がシステム上で評価しコメントを付与することができる。加えて、解答の評価を支援する機能として、教員がプログラムに対する入力と出力の組（テストケース）を複数設定すると、それぞれの解答についてどのテストケースと合致したかを自動的に判定する機能を有する。例えばテストケースとして正しいプログラムにおける入力・出力の組や、典型的な誤りを含むプログラムにおける入力・出力の組などを設定すれば、それぞれの解答の動作を教員の目で詳細に確認することを大幅に省略することが可能となり、解答の評価を効率的に行うことができる。

5

自己調整学習がプログラミング演習に与える影響

近年、「自律的な学び」あるいは「学び方の学び」が注目されている⁽³⁰⁾。自己調整学習（SRL: Self-Regulated Learning）⁽³¹⁾は、そのための一つの理論体系である。知識だけでなく、学習を促進するための学習方略、自分の理解状態を確認するメタ認知、学習目標達成に向けた自己効力感などを包括的に捉え、学習者自らがこれらを意図的に制御することを重視している。

プログラミングでも、受け身に講義を受講するだけではなく、教授された知識を能動的に確認・活用することの重要性は従来から指摘されている⁽³²⁾。また、大学入学者にはプログラミング経験がないものも多数含まれ、過去の学習経験における学習方略やメタ認知技能などを新たな学習主題に転移させて活用する必要がある。

本論文の第一著者は2017年度に現所属に異動し、2018年度から学部1年生前期のプログラミング演習授業を担当している*。一方、第二著者は、2015年度から同じく1年生前期にSRLを促進するための授業を実施していた。そこで、この授業

* 内容は前述のものに配列を加えたもので、並べ替えや探索を扱う代わりに、プログラム動作をトレースする課題で、基礎的概念の理解確認、問題解決に至る分解・整理の方法の習得を図っている。

が、プログラミング演習授業の受講意識に与える影響について調査・分析を行った⁽³⁰⁾。

5.1 SRL とプログラミング

SRL は、予見、遂行、自己内省の各段階から成る個人的フィードバックループとしてモデル化される⁽³¹⁾。予見段階は、学習に先立って、自己調整の準備と自己動機付けを行う段階である。遂行段階は、実際の学習や課題解決に対応し、セルフコントロールと自己観察から成る。自己内省段階は、学習や課題解決の結果を自己評価し、その結果の原因を自身の能力、努力、方略に帰属させ、次の学習の予見段階への結び付けを行う。

プログラミングの学習においても、処理の「意味」を考え、学んだ知識を能動的に新たな課題に活用することが求められ、SRL は重要な役割を果たす。上記のフィードバックループは、課題を与えられてからプログラムを完成させるまでの過程に対応する、と考えると分かりやすい。例えば、予見段階では、課題の目標設定や課題を解く学習方略の計画を行う。つまり、プログラムの完成形をイメージし、そのために自身の有するプログラミング技法をどう適用すればよいかの見通しを立てる。この見通しが明確な学習者は課題に対する自己効力感が高まり、不明確な学習者は自己効力感が低いま行き当たりばったりでプログラムを書き始めてしまう。次の遂行段階は、実際にプログラムを作成する過程に対応する。SRL の重要な学習方略の一つに「イメージ化」があるが、これはプログラミングでは、例えば、静的なコードが動的にはどのような動作になるかを、図などの表象でイメージすることを意味する。また、自分の認知過程に対するメタ認知も重要である。これには、自分が作成しているプログラムの意味をしっかりと理解できているかどうかを自らに問い掛けるセルフモニタリングなどが含まれる。最後の自己内省段階は、結果の振り返りを行う段階である。例えば、テストの結果が悪かった原因が、アルゴリズムの理解不足なのか、文法知識の不足なのか特定できれば、次に向けた学習方略の適応的計画が可能になり、自己効力感を維持できる。しかし、そもそも自分の学習能力が不足している、と判断してしまうと、具体的な方略が計画できず、自己効力感も低下して学習回避的行動につながってしまう。

5.2 SRL の授業の目的と構成

SRL に関する教育実践事例の多く⁽³¹⁾は、数学や英語など特定の科目の中で学習計画を立案させるなど、

前節で述べた SRL の各段階を学習者に意識させる直接的な働き掛けを行うものである。しかし、本実践の授業は、特定の科目には依存せず、学習者が SRL の理論を自らの学習経験と結び付けて内省・概念化し、以後の学習活動におけるメタ認知や学習方略の意識的な活用を促進することを意図している⁽³⁰⁾。大学生は既に多くの学習経験を有しているはずであり、その経験が SRL の観点から解釈できることに気付かせ、以後の学習行動の意識的修正を促すことを目的としている。

そのために、授業設計にはドキュメンタリービデオとオンラインレポート提出を組み合わせた枠組み⁽³³⁾を適用している。この枠組みでは、構成主義的な学習観に基づき①学習者の既有知識・経験の活用、②主題に関する真正な状況・文脈の提示、③他者と自らの考えを対比する機会の提供、という方針を取る。

授業は 4 こまの構成である。SRL を含む体系的知識の説明、ビデオ視聴、レポート提出と配布を行う。ビデオは 25 分程度のものを 2 本視聴させる。その一つは、数学に対して自己効力感のない中学生に対して塾講師が自律的な学習目標設定を促し、中学生が明確な目標を持って適応的に学習するようになるまでの様子を描いている。このビデオについて、「塾講師が女子生徒の学習を促進するために行っていることを、自分の学習経験と結び付け、SRL の観点から説明せよ」（「女子生徒」は、上述の「中学生」と同一人物である）というレポート課題を課す。次の授業で、提出されたレポートを全員に配布し、教員が内容を適宜紹介して、自他の考えを比較・吟味させる。その上で、更に踏み込んだレポート課題として、「ほかの人のレポートも参考にした上で学習理論・動機付け・自己学習の概念を用いて、学習が促進されなかった要因、指導者が学習者に対して行っていた働き掛けとその意図、指導者が行った働き掛けの効果とその理由」などを問う課題を提示して、同じビデオを視聴後、次の授業で再度レポート配布、紹介を行う。授業後に事後アンケートを行う。

5.3 調査方法

プログラミング演習受講意識の調査には Motivated Strategies for Learning Questionnaire (MSLQ)⁽³⁴⁾を用いた。MSLQ の下位尺度は「自己効力感」、「内発的価値」など五つである。今回は、「その科目で教わる内容を、必ず理解できると思う」など 9 項目から成る自己効力感と、「勉強するとき、重要な事柄を自分の言葉に置き換える」など 13 項

目から成る認知的方略の使用について調査した。

同曜日の午前に SRL 授業を含む専門導入科目、午後にプログラミング演習授業が開講されており、表 2 に示すスケジュールで調査を行った。プログラミング演習授業では第 9 週までに基本的な概念の導入と、各週で導入された概念の理解を確認するためのプログラム作成演習を実施した。第 10～13 週では全ての概念を用いたプログラムの読解・作成演習を実施した。第 14、15 週には期末試験と授業のまとめを行った。この中で、MSLQ 調査を、入学直後（第 1 週）、SRL 授業実施前（第 9 週）、実施後（第 13 週）、期末試験実施後（第 15 週）の計 4 回実施した。なお、プログラミング演習授業内では、SRL を促進するような特別な働き掛けは行っていない。

SRL 授業では、第 1 週目に入学時アンケートを実施し、プログラミング経験の有無などについて調査を行った。事後アンケートから授業の主観的理解度を求めた。アンケートは、ビデオ、レポート、授業全般に関する設問 31 問で構成されている⁽³⁵⁾。因子分析を行ったところ、第一因子が「大学での学習を進める上で参考になった」、「学び方に関する考え方が変わった」、「ビデオを二度視聴して理解が深まった」、「ほかの人のレポートを見て理解が深めることができた」など、授業全体やビデオ、レポートによる学習主題の理解に関わる項目を含んでいたため、これを授業の主観的理解度とした。

5.4 調査結果

レポート・アンケートを全て提出した 123 名の学習者を、前述した授業主観的理解度の高低と入学前のプログラミング経験の有無によって群に分け、2×2×4 混合計画の分散分析を行った。

図 3 の自己効力感については、学習者全体に、入学時から SRL 授業前の間に値が有意に低下し、SRL 授業後、テスト後と順次上昇していた。これは、自己効力感が、プログラミング演習前半で低下し、SRL 授業によって向上に転じた、と解釈することができる。

図 4 の認知的方略の使用については、プログラミング経験有群、及び、授業主観的理解度高群で、SRL 授業後からテスト後にかけて、値が有意に上昇した。これは、SRL 授業によって、学習者が、イメージ化やメタ認知などの認知的学習方略を意識し、プログラミング演習のテストの準備・受験を通じて、これらの学習方略を活用した結果、と解釈することができる。また、今回詳しく確認していない

表 2 授業・調査のスケジュール

週	専門導入科目	プログラミング演習授業
1	入学時アンケート	MSLQ1
9	事前アンケート	MSLQ2
10～12	SRL 授業	
13	事後アンケート	MSLQ3
14		期末試験
15		MSLQ4

図 3 自己効力感の変化（7 件法）

図 4 認知的方略の使用の変化（7 件法）

が、SRL 授業後からテスト後の認知的方略の使用と自己効力感の間には有意な相関が見られ、互いの向上が関係していることが考えられる。

一方、プログラミング経験無群、及び、授業主観的理解度低群には、認知的方略の使用の向上は見られなかった。特に、授業主観的理解度の高低群の間には、一貫して認知的方略の使用に差が見られる。元々認知的方略の使用にたけた学習者は、SRL 授業をよく理解できたと感じ、プログラミング経験がなくても SRL 授業で意識した学習方略をプログラミング演習で活用することができた可能性がある。

6 まとめ

筆者はプログラミングの導入授業をアルゴリズム的思考の教育と位置付けて実践を行ってきた。また、各年度の実践における学習効果の検証に基づいて授業の改善や学習支援システムの開発・導入を行った。その結果、システムの導入後の学習効果が有意に向上したことが分かった。また、2018年度には新たな取組みとして、自己調整学習について学ぶことがプログラミング演習授業における学習活動に及ぼす影響についての調査を行った。その結果、入学前にプログラミング経験を有する学習者や、自己調整学習に関する授業への主観的な理解度が高い学習者において、自己効力感や認知的方略の使用に良い影響があることが分かった。

今後も学習効果の確認を行った上で、プログラミング演習授業及び授業支援システムの改善を行う必要がある。また、プログラミング経験のない学習者や主観的な理解度が低い学習者についても、自己効力感や認知的方略の使用について向上するような取組みを行いたい。

謝辞 本研究はJSPS科研費JP19300285, JP22300286, JP23501103, JP26350284, JP26560127, JP15K01023, JP16H03074, JP18H03346の助成を受けたものです。

本論文で取り上げた実践に初期から共同で取り組んできた、信州大学の不破 泰教授、香山瑞恵教授、新村正明准教授に感謝申し上げます。

文献

- (1) 國宗永佳, 香山瑞恵, 新村正明, “情報工科学学生に対するアルゴリズム的思考法教育の実践と評価 (第2報),” 教育システム情報学会研究報告, vol.27, no.6, pp.177-182, March 2013.
- (2) 國宗永佳, 山本 樹, 新村正明, 香山瑞恵, “初級プログラミング学習におけるプログラム作成過程と支援方法の検討,” 教育システム情報学会第38回全国大会講演論文集, G5-1, pp.379-380, Sept. 2013.
- (3) J. M. Wing, “Computational thinking,” Commun. ACM, vol.49, no.3, pp.33-35, March 2006.
- (4) J. M. Wing, “計算論的思考,” 情報処理, vol.56, no.6, pp.584-587, May 2015.
- (5) 仲林 清, 國宗永佳, “自己調整学習に関する内省・概念化を促す授業がプログラミング演習科目の受講意識に与える効果,” 日本教育工学会研究報告集, vol. JSET19-1, pp.463-470, March 2019.
- (6) L. N. ラング, アルゴリズムの思考方法 上・下, 明治図書出版, 東京, 1970.
- (7) 駒林邦男, “教授における数学的論理学の適用の試み, アルゴリズムの教授を中心として,” 日本教育学会大会研究発表要項, vol.22, pp.30-31, Aug. 1963.
- (8) 宮坂瑠子, “教授過程における学習行為の組織化 - とくにアルゴリズム化について,” 日本教育学会大会研究発表要

- 項, vol.26, p.25, Aug. 1967.
- (9) 小菅貴彦, “基本アルゴリズムを学習するエデュメント教材,” 信 学 技 報, ET2005-110, pp.17-20, March 2006.
- (10) 宮田 仁, 大隅紀和, 林 徳治, “パソコンを利用した「ハノイの塔」における問題解決過程の分析,” 日本教育情報学会学会誌, vol.11, no.2, pp.13-20, July 1995.
- (11) 阿部哲也, “アルゴリズム教育のための教材の開発,” 日本教育工学雑誌, vol.27, no.Suppl, pp.45-48, March 2004.
- (12) 梅田恭子, 江島徹郎, 野崎浩成, “高等学校普通教科「情報B」におけるアルゴリズム学習教材の開発と評価,” 愛知教育大学研究報告 (教育科学編), vol.55, pp.167-173, March 2006.
- (13) 和田 勉, “問題解決から導くプログラミング教育方法の案,” 情処学コンピュータと教育研究会研報, vol.98, no.102, pp.57-64, Nov. 1998.
- (14) 杉浦 学, 松澤芳昭, 岡田 健, 大岩 元, “アルゴリズム構築能力育成の導入教育, 実作業による概念理解に基づくアルゴリズム構築体験とその効果,” 情処学論, vol.49, no.10, pp.3409-3427, Oct. 2008.
- (15) 永野和男, 沢田吉宏, 三輪吉和, “情報処理教育のためのローカル・エリア・ネットワーク・システムの開発, アルゴリズム教育用構造化 BASIC とその知的エディタ,” テレビ学技報, vol.9, no.34, pp.29-34, 1985.
- (16) 小西達裕, 鈴木浩之, 伊東幸宏, “プログラミング教育における教師支援のためのプログラム評価機構,” 信学論 (D-I), vol.J83-D-I, no.6, pp.682-692, June 2000.
- (17) 不破 泰, 國宗永佳, 香山瑞恵, 新村正明, 宮尾秀俊, “情報工科学学生に対するアルゴリズム的思考法教育手法の提案と実践,” 教育システム情報学会研究報告, vol.23, no.6, pp.34-41, March 2009.
- (18) kujirahand, “日本語プログラミング言語なでしこ,” <https://nadesi.com/top/> (閲覧日 2019 年 3 月 29 日)
- (19) 不破 泰, 國宗永佳, 香山瑞恵, 新村正明, 宮尾秀俊, “情報工科学学生に対するアルゴリズム的思考法教育の実践と評価,” 信学技報, ET2009-49, pp.51-56, Nov. 2009.
- (20) 國宗永佳, 山本 樹, 不破 泰, 香山瑞恵, 新村正明, “情報工科学学生に対するアルゴリズム的思考法教育手法の改善,” 教育システム情報学会研究報告, vol.25, no.6, pp.55-60, March 2011.
- (21) 國宗永佳, 山本 樹, 香山瑞恵, 新村正明, 不破 泰, “情報工科学学生に対するアルゴリズム的思考法教育の実践,” 教育システム情報学会第36回全国大会講演論文集, pp.444-445, Aug. 2011.
- (22) 佐藤隆博, S-P 表の入門, 明治図書出版, 東京, May 1985.
- (23) 佐藤 亮, 香山瑞恵, 國宗永佳, 伊東一典, 橋本昌巳, 大谷 真, “アルゴリズム的思考法の学習における課題の適切性に関する検討,” 信学技報, ET2012-43, pp.31-36, Oct. 2012.
- (24) 小林 慶, 國宗永佳, 香山瑞恵, 新村正明, “アルゴリズム的思考法教育を支援するビジュアルプログラミング環境の開発,” 教育システム情報学会研究報告, vol.27, no.4, pp.3-8, Nov. 2012.
- (25) 岡本雅子, 喜多 一, “プログラミングの「写経型学習」における初学者のつまづきの類型化とその考察,” 滋賀大学教育学部附属教育実践総合センター紀要, vol.22, pp.49-53, March 2014.
- (26) 中西 渉, 辰巳丈夫, 西田知博, “PenFlowchart によるプログラミング導入教育の評価,” 情処学研報, vol.2013-CE-121, no.9, pp.1-7, Oct. 2013.
- (27) Lifelong Kindergarten Group at the MIT Media Lab,

- “Scratch”, <https://scratch.mit.edu/> (閲覧日 2018 年 11 月 7 日)
- (28) J. Maloney, L. Burd, Y. Kafai, N. Rusk, B. Silverman, and M. Resnick, “Scratch: A sneak preview”, Proc. of Second International Conference on Creating, Connecting and Collaborating through Computing, pp.104-109, July 2004.
- (29) Google, “Blockly”, <https://developers.google.com/blockly/> (閲覧日 2018 年 11 月 7 日)
- (30) 21 世紀型スキル：学びと評価の新たなカタチ, P. グリフィン, B. マクゴー, E. ケア (編), 北大路書房, 京都, April 2014.
- (31) 自己調整学習—理論と実践の新たな展開へ—, 自己調整学習研究会 (編), p.13, 北大路書房, 京都, April 2012.
- (32) 森山 潤, “中学生の Logo プログラミングにおける課題の拡散性と思考過程の関連,” 教育システム情報学会誌, vol. 17, no.1, pp.3-13, April 2000.
- (33) 仲林 清, “組織における問題解決を主題とするビデオとオンラインレポートを活用した授業実践,” 教育システム情報学会誌, vol. 32, no.2, pp.171-185, April 2015.
- (34) P. R. Pintrich and E. V. De Groot, “Motivational and self-regulated learning components of classroom academic performance,” Journal of Educational Psychology, vol.82, no.1, pp.33-40, March 1990.
- (35) 仲林 清, “自己調整学習を主題とするビデオとオンラインレポートを活用した授業実践における学習者の意識変化,” 日本教育工学会研究報告集, vol.JSET16, no.4, pp.39-46, Nov. 2016.

(2019 年 2 月 5 日受付, 4 月 10 日再受付)

國宗永佳 (正員：シニア会員)

2000 信州大大学院工学系研究科博士前期課程情報工学専攻了。2003 同研究科博士後期課程システム開発工学専攻了。博士 (工学)。2003 信州大・工・特別研究員, 2004 同助手, 2007 助教を経て, 2017-04 千葉工大・情報科学・教授。情報処理学会, 教育システム情報学会, 日本教育工学会, IEEE 各会員。教授・学習支援システムに関する研究に従事。特にプログラミングをはじめとする情報技術教育・学習の支援に興味を持つ。

仲林 清 (正員)

1982 東工大大学院理工学研究科修士課程了。同年日本電信電話公社 (現 NTT) 入社。同社情報通信処理研究所主幹研究員, 放送大教授を経て, 現在, 千葉工大・情報科学・教授。熊本大・教授システム学専攻・客員教授。教育システム情報学会会長, 日本イーラーニングコンソシアム副会長, 人工知能学会, 日本教育工学会, IEEE 各会員。ICCE2009 Best Technical Design Paper Award 受賞。2006 早大大学院人間科学研究科博士課程了。博士 (人間科学)。