例題を手がかりにしたプログラミング基本知識の準備

土屋孝文*1・高井峻太

Email: tsuchiya@sist.chukyo-u.ac.jp

*1：中京大学工学部

◎Key Words アルゴリズム、プログラミング，学習支援

1. はじめに ー プログラミング支援に向けて

本研究は情報系基礎科目「アルゴリズムとデータ構造」を対象に、学習支援環境の開発と運用を行っている。この実践では、先輩にあたる学生が自分自身の学習経験に基づき後輩たちの学習支援環境設計に関わるサイクルを継続している。

この科目の学習には 3 種の領域知識、すなわち、対象問題のアルゴリズムの理解、アルゴリズムに対応するコーディング（プログラム生成）、プログラムの実行過程とアルゴリズムとの対応（プログラム解釈）が相互に関係している。本研究では各領域に有効な支援環境の設計と運用を検討してきた。

アルゴリズムの理解に関する環境については、対象問題に対する学習者自身による手続き説明から始め、協調的活動を利用して集団内の異なる解（思考）を比較の後、理論的解答への橋渡しまでを一般的なフレームワークと考え、対象問題に応じた具体的な支援環境の設計への適用を試みている。

整数 N 以下の素数を求めるアルゴリズムを例にすると、学習者はまずウェブページを用いて整数 N 以下の素数を選びだすまでの解答時間を競いあう（図 1 上）。ここから自分の解法を説明、共有、比較する活動が続く。

毎年の運用では、解法は主に(A) 対象の数について、それまでに確定済の素数で割りきれるかを判定する方法、(B) 素数を確定した場合、それより後ろの数をその素数で割りきれるかを判定していき、候補リストから消去する方法、(C) 素数を確定した場合、それより後ろの倍数を全て消去する方法に分散する。電子掲示板によってこれらの解法を比較したあと、図 1（下）に示すアニメーションから、素数確定の範囲を√N までに限定した(C)のアルゴリズム（エラトステネスのふるい）を理論的解答として確認する。言語による解法理解は一定の結果を示す。

素数アルゴリズムの学習では、アルゴリズム理解の活動と並行しながら、自分のアルゴリズムをコーディングし(プログラム生成)、3 種類の標準コードを比較理解する活動（プログラム解釈）を行う。アルゴリズムの理解と比較して、言語表現をプログラムと対応づける活動には学習者に一定の難しさがみられる。本稿は、素数アルゴリズムの学習場面を例に、例題プログラム集を学習資料として活用するプログラム生成と解釈の学習支援について報告する。

図 1 自分の解法を確認するための作業ページ（上）と理論的解答を確認するアニメーションページ (下)

2. 例題集の活用

プログラム生成の支援には、基本的な問題処理知識に対応する、小さく具体的な例題プログラム集を準備し、一旦これらを学習した後の学習者に、課題に応じた適切な例題の利用を促す学習環境を運用してきた[2]。学習者が例題の知識を十分内化できていない場合は例題集に戻って想起や再学習が可能な環境である。表 1 はバブルソートなどの基本ソートをプログラムするレベルまでに必要と考えられる 9 つのカテゴリと例題である。例題には、適度な大きさであること、適当な量の基本知識の具体例になって

表 1 基本例題集

	カテゴリ	例題 1	例題 2
1	入出力	おうむ返し	配列格納
2	条件分岐	偶数/奇数判定	月 ⇒ 季節
3	ループ	N 個合計	N 個/終了キーまで合計
4	配列処理	要素合計	連続要素縮約
5	文字列	長さ	1 文字検索
6	二重ループ	九九表	一次元配列 ▽型出力
7	その他（基本処理）	配列要素交換（swap）	乱数ライブラリ
8	関数	sum 関数	配列要素の合計
9	再帰	階乗	二分探索

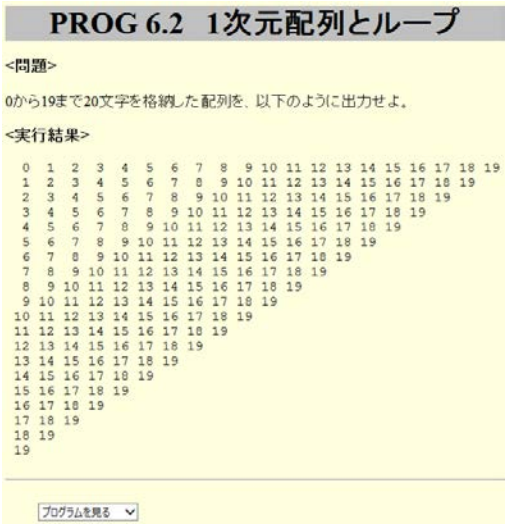

図２　二重ループ処理の基本例題

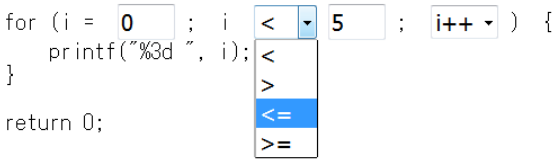

図３　for ループの実行確認　設定例

いること、学習者自身がいつでも例題コードに立ち戻って学習できること（学習者自身で一定の抽象的知識を得られる良質な例題であること）などの要件が求められるだろう。

素数に関するアルゴリズムをコーディングする学習者には、一次元配列を二重ループで扱う例題（図2）の利用をヒントに示す。例題は入出力例とプログラムから構成される。これまでの運用からは、一定の学習者が課題中に例題を参照し、その有用性に肯定的な評価がなされている一方、例題を利用した課題プログラム作成の難しさも確認されている。

江川ら（2004）は、二重ループの例題プログラムに関する不完全な説明事例を収集し、その困難さについて、単により基本的な構文理解を積み上げる際の準備不足とみるのではなく、入出力だけに注目した表層的理解や、プログラム実行に関する基礎的な知識が不十分なままの学習が影響しており、実際の学習を考慮した多様な支援環境の必要性を指摘している[(1)]。

そこで今年度の実践では、学習者に例題プログラム中の操作や実行順序（制御情報）と使用されている変数の役割に注目した解釈を促すため、プログラム中の変数の初期値や構文の一部を変更しながら、実行結果を繰り返し確認できる機能[(3)]を例題集に導入した。 図３は、for 構文の実行を確認する例題ウェブページにおける設定の例である。設定後は、JavaScript のイベント処理を用いて、ウェブページ上のプログラムと同じ計算を行い、プリント文の出力を生成、表示する。

3.　運用例と今後の課題

表１の例題集や課題プログラム解答への実行機能の追加のほか、例題集に基本構文形式と実行結果を確かめるだけのより入門的な例題（while, for, if, switch, 配列, 関数）を追加した。このような基本知識は、長期休暇明けなど、一時的に不安定になったプログラム知識を確認する手がかりに有用と考えられる。基本知識の確認については、85.7%（N=91）のユーザより有用との評価を得た。

図４は素数を求める３種類のプログラムを実行結果から比較するページの出力例である。プログラム中のマクロ N を設定し、実行プログラムを選択することによって、二重ループによる実行の様子をプリントする。プログラムの比較確認については 92.6%（N=68）より有用との評価を得た。

プログラム実行順序や変数の状態を確認するための基本機能の追加が今後の課題である。

参考文献

(1) 江川紘美, 三宅芳雄：”プログラミング学習の認知過程 - 基本スキーマと多重な認知構造に基づく解明 - “, 日本認知科学会第 21 回大会論文集, pp. 162 -163（2004）.
(2) 土屋孝文, 齋藤真琴, 鹿内拓哉, 原田翔一, 松井浩紀, ”例題を利用するプログラミング支援環境-基本アルゴリズムの学習場面を例に-”, PC カンファレンス論文集, pp. 9-10（2015）.
(3) 占部弘治：”プログラミング科目の理解を助けるインタラクティブ教材の検討”, PC カンファレンス論文集, pp. 63-64（2016）.

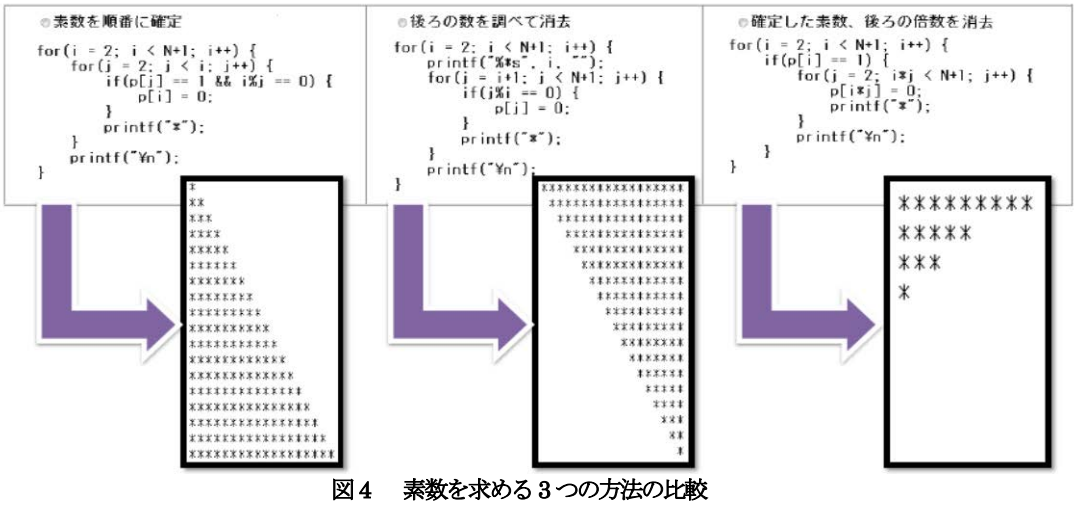

図４　素数を求める３つの方法の比較