なぞれるけれどプログラムにできない　- コーディングの支援に向けて

土屋孝文*1
Email: tsuchiya@sist.chukyo-u.ac.jp

*1：中京大学工学部

◎Key Words　　学習支援，プログラミング

1.　はじめに

本研究は C 言語入門科目に続く情報系基礎科目「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。履修者の多くは、プログラミングの学習初期にあたり、カリキュラムを通じて、不完全で不安定な知識を繰り返し再学習しながら、次第にプログラミングスキルを熟達させていく。授業は大まかに 3 種のフェーズ、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（解答例となるプログラムの理解）」の順に進行する。図 1 に示すように、学習者はカードや表などの自然な情報表現（外部データ表現）を適切に操作すること（アルゴリズムをなぞること）から始めて、対応するプログラムの生成（修正）を行う。これまで本研究は、この過程にへだたりを覚える学習者に向けて、基本知識の再確認、既有知識の活性化や誘発（想起や利用）、誤概念の修正、問題解決（プログラム生成）に関するガイドなど、（再）学習を通して先に進むための外的支援ツールを検討、提供してきた。もちろん一定の支援は期待できるが、これらは学習者側からツールにアクセスすることで一様に「利用可能」なだけである。対話的に説明を引き出して、各自が理解を得たり修正したり深めたり、問題解決を進めることはできない。

そこで本稿では生成 AI （ChatGPT） との対話による学習の可能性を検討する。以下では、選択ソートの各フェーズを例に、ChatGPT を、現在の学習環境にどのように取り入れられそうか、また、これまでの（再）学習支援ツールと ChatGPT の利用との関係について述べる。

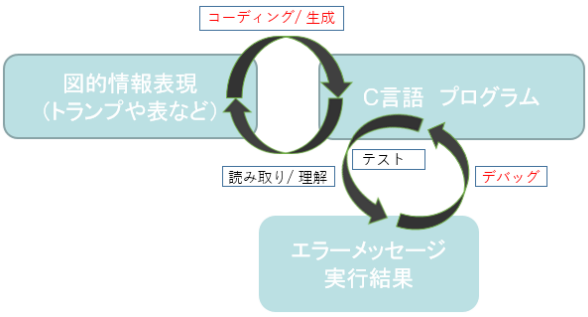

図 1　3 種類の情報表現と表現に対する操作

2.　アルゴリズムの理解 – 説明生成

アルゴリズムの形式的な説明の前に、操作の具体事例から帰納的な推論による手続きの説明（仮説生成）を促すこととしている。具体的な図的情報表現（トランプ）を正解アルゴリズムにしたがって自分で操作する、いわば「アルゴリズムを手作業で正しくなぞってみる」ツールを提供する。図 2 は選択ソート用ツールの観察部である。

ツールは、まず、典型的な入力（[3, 5, 4, 2, 1]）の操作事例のほか、バブルソートと誤りやすい入力（[5, 1, 2, 3, 4]）やユーザ自身の入力について、順に操作事例を表示する。

次に、学習者は与えられた入力に対する操作を試みて、自分が考えた手続きを検証する。手続きを電子掲示板に共有したあと、最後に教員が講義を通して正解を示す。

正解の手続きが確認されたあとに、説明コメントを付加したプログラムと実行結果を例示するレポート作成へ進む。履修者は教科書やネットなど、多くの資料を利用し解答を選択できるが、「なぞれるけれど、自分だけではプログラムにはできない」という振り返りが多い。これは、基本的な既有知識を組み合わせて利用しながら、手続きをアルゴリズムとしてとらえ、プログラムを生成していく方略知識が不十分な状況と考えられる。

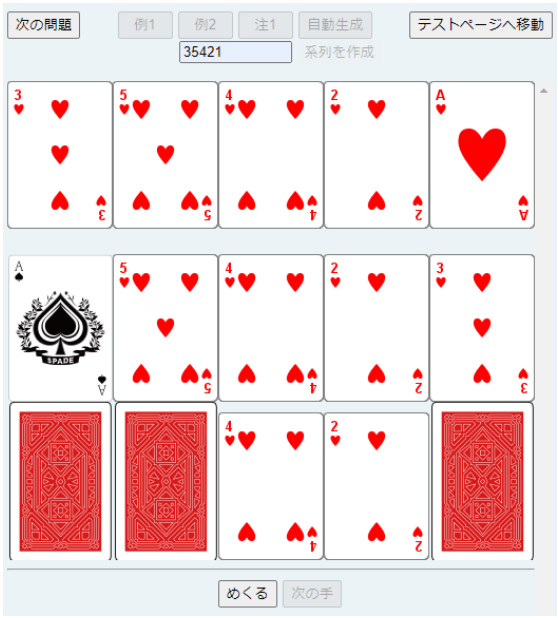

図 2　選択ソートアルゴリズムのなぞり

3.　コーディング – 基本知識の確認

プログラム生成の支援に、まず、C 言語の構文知識や定型的な処理パターンを再学習するための例題プログラム集を準備し、課題に応じた例題の参照や利用を促す。表 1 は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、小さくて覚えやすく、多くのプログラムの部品となる操作や定型的処理を含む題材を検討している。

例題の再学習には、例題プログラムを表面的にながめ

るだけではなく、プログラムの一部を再生してみたり（写経）、正しく並び替えたりするツールを提供した。図3は、一次元配列の要素を二重ループ処理で操作する定型的な例題（◤型出力）である。ツールは、入力文字への正誤判定を通して、基本構文の組み合わせによる例題構成の確認を促す。

基本ソートは、一次元配列の二重ループ処理と要素交換の例題を適切に変形し組み合わせようとすれば、プログラムの大枠（不完全な部分コード）が得られる。選択ソートでは、「先頭と最小値の要素交換」の繰り返し手続きのコーディングにこの大枠が利用される。

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

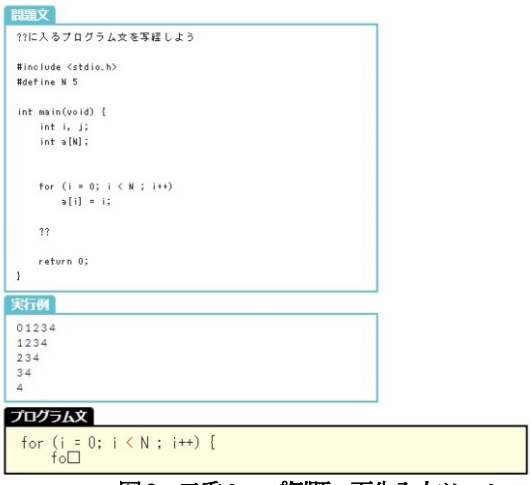

図3　二重ループ例題　再生入力ツール

4.　コーディングとプログラム理解 － ChatGPT

外的な図的情報表現上の手続きをプログラムに対応づけるには、大きく 2 つの視点が必要と考えられる。１つはデータの表現や制御に使用される変数、定数およびデータ構造の設定である。選択ソートの図的情報表現（トランプ列）は配列で表現され、最小値の決定や配列操作には変数が必要となる。もう１つは、視覚や手指で行っている処理を適切に一連のプログラムに対応づけることである。選択ソートでは、最小値の発見と交換の繰り返しの手続きを二重ループのブロックに対応づける必要がある。

このようなコーディング方略の獲得支援は容易ではない。本研究ではこれまで、手続きのなぞりとプログラムとの間に、コンピュータ側の操作にあたる中間的なアルゴリズム表現を提供する支援を検討してきた。今回は、レポート課題の初期プログラム（基本的なデータ表現とテンプレートプログラム、コメント文を含む）からのプログラム完成と説明を ChatGPT に求めることとした。プロンプトは「{初期プログラム}は途中まで作成した選択ソートのプログラムです。ステップバイステップでプログラムを完成させてください」である。

応答の説明文にはコーディング方略のヒントが含まれている。また、説明文読解には、先行するアルゴリズムの理解と基本知識の確認が有効と考えられる。さらに個別の説明にはプロンプト「{プログラムブロック}の部分をもう少し詳しく説明してください」が有効である。

5.　テストとデバッグ － ChatGPT

選択ソート課題には、「先頭と、より小さい値の要素交換の繰り返し」を繰り返す、誤ったプログラムの提出がみられる（図 4）。そこで、観察ツールでは暫定最小値が先頭に移動しないことを示すようにした（図 2）。

```
mis-selection-sort() {
    for (i = 1; j =< n - 1; i = i + 1){
        min ← i;
        for (j = i + 1; j =< n; j = j + 1){
            if (A[j] < A[min]){
                swap(A[j], A[min]);
            }
        }
    }
}
```

図4　誤りのある疑似コード

課題後の授業では、図 2 のようにトランプを用いた手続きと、誤答コードの操作を対応づけながら、正解と誤答を比較する。解説後に振り返りの質問を行ったところ、「なぞりは正解したが、誤った手続きのコードを書いた」という選択肢は 23%（前年とほぼ同じ)で、講義後は誤りの納得や修正を期待できる。一方、図の工夫や提出プログラムに説明コメントを求めるだけでは、誤りの気づきを誘発しにくい事例である。そこで、ChatGPT に「{作成プログラム} は選択ソートのプログラムです。改良点をステップバイステップで教えてください」をプロンプトとすると、正しく誤りを指摘し修正点を示す返答が得られた。

6.　おわりに

ChatGPT からの情報の引き出しと対話を通した理解を支援するという立場から、教材やツールを検討したい。授業内で利用方法の共有や議論を進める予定である。

本授業レベルのレポートは「とにかく答えを生成する」活動から「答えや生成過程を理解する」活動に変わり、答えの質を評価する意味は少なくなるだろう。むしろ、課題中の ChatGPT 利用前後の理解の変化に注目させる、いわゆる振り返り（リフレクション）が重要と考えられる。