# プログラミングにおける構造的理解のための部品の段階的拡張手法を用いたシステムの評価

# Evaluation of a System with Expandable Modular Statements for Structural Understanding in Programming

古池 謙人[*1], 東本 崇仁[*2], 堀口 知也[*3], 平嶋 宗[*4]
Kento KOIKE[*1], Takahito TOMOTO[*2], Tomoya HORIGUCHI[*3], Tsukasa HIRASHIMA[*4]
[*1] 東京工芸大学大学院工学研究科
[*1]Graduate School of Engineering, Tokyo Polytechnic University
[*2] 東京工芸大学工学部
[*2]Faculty of Engineering, Tokyo Polytechnic University
[*3] 神戸大学大学院海事科学研究科
[*3]Graduate School of Maritime Sciences, Kobe University
[*4] 広島大学大学院工学研究科
[*4]Graduate School of Engineering, Hiroshima University
Email: m1865005@st.t-kougei.ac.jp

**あらまし**：プログラミングではソースコード全体が果たす機能のみならず，その全体を構成する個々の機能を部品として認識することで，他のプログラムへの部分的な再利用が望める．こういった構造的理解を目的として，著者らはこれまでに部品の段階的拡張手法を提案し，システムを用いて支援を試みてきた．本稿ではさらに，本手法による効果とシステムのフィードバックによる効果を調査するためにシステムを用いて実験を行い，その結果を報告する．
**キーワード**：プログラミング学習，部品の段階的拡張手法，構造的な理解，学習支援システム

## 1. はじめに

プログラミングでは if 文や for 文などの構文を組み合わせて入力データを操作し，期待する出力を行わせることができる．しかし「入力 A と入力 B を足して出力する」ということを行う極めて単純なプログラムでも，「足す」構文と「出力する」構文の２つが必要となる．より複雑なプログラムになれば，より多くの構文を要することは自明である．では，複雑なプログラムを記述する時に熟達したプログラマが構文を一つずつ考えながら記述しているかということについては，そうではない(1)．例えば，熟達したプログラマがある配列を並べ替えたいと考えた時には，2 重 for 文を容易に呼び出すことができる．これは，熟達したプログラマが配列の並べ替えに必要な連続した構文を並べ替えの機能と関連付けて理解し，それを再利用してきたからである．プログラミングにおいては，連続した構文を機能として理解することで他のプログラムに再利用することが可能となり，さらに，複雑な構文からなるプログラムを複数の機能として理解することが可能になる．本研究では，こういった機能として有意味な連続した構文を部品と定義している．

学習者が部品を獲得するためには，連続した構文を明示的に部品として再利用することが有用である．これにより，学習者は部品の機能が何であるかを注意深く考えながらプログラムする必要が生じる為，部品の機能としての理解が促進される．しかし，通常のプログラミング教育においては機能と連続した構文を結びつけた部品的な教示は行われるものの，学習者に明示的に部品を再利用させる機会は少ない．

よって，著者らはこれまでにプログラムを部品として獲得することを目指して学習手法を提案し，支援システムの開発を行い有効性の検証をしてきた(2),(3)．しかし，提案手法が有効であったのか，システムのフィードバック（以降，FB と表記）が有用であったのかは明らかではない．よって本稿では，システムの FB の有無による学習効果の違いを調査した．

## 2. 部品の段階的拡張手法とシステム

1 章では，機能を持つ連続した構文を部品として理解することで他のプログラムに再利用して活用することができたり，複雑なプログラムを読解する助けになったりするという利点を主張した．そこで本章ではこれまでに著者らが提案してきた連続した構文を部品として獲得するための部品の段階的拡張手法を説明し，本手法を用いたシステムを紹介する(2)．

本手法では，まず，学習者に作るべき機能が提示され，学習者はプログラムにおける 1 行ずつの構文から有意味な塊ごとに部品の構築を行う．その後，以前に作った機能を包括した機能が作るべき機能として提示されるので，学習者は既に自身が構築した部品を再利用し，そこに新たな構文や既存の部品を加えることでより大きな部品に変化させる．こうして自身が構築した部品を再利用するには，その部品の機能が何であるかを注意深く思考する必要がある．また，部品を変化させることを繰り返し，部品の構

造を段階的に拡張することで，学習者に部品の機能的な理解だけでなく，部品同士の関係性を理解させることを促すことができると著者らは考えた．

開発したシステム（図 1）は，本手法に沿ってより小さい機能から段階的に課題を提示する．また本システムは解答時の正誤判定機能や最上流の誤り箇所を指摘する機能，その正答を提示する機能といった FB 機能を 3 種類持っている．本稿は本システムの FB 機能の有無による学習効果の違いを検証した．

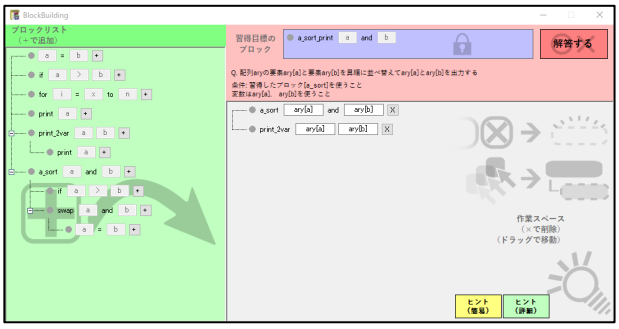


図 1 システムインタフェース

## 3. 評価実験

これまで本研究では，本システムに学習効果があることを検証し[2]，紙教材学習との比較では有意な学習効果が得られている[3]．しかし，提案手法を用いた構造化の作業が有効であったのか，もしくはシステムが行った FB が有効であったのかは明らかではない．よって本実験では，システムにおいて FB の有無による学習効果の違いを調査した．

### 3.1 実験概要

被験者はプログラミングの講義で 3 年間プログラミングを学んだ大学生 15 名であり，for 文，if 文など基本的な構文や，配列の並び替えなどのアルゴリズム，関数の学習については修了しているものである．また，本実験の対象言語は C 言語を基にポインタや関数マクロの概念を省略したものとなっている．

事前では，系列的に発展が可能な問題を記載したテストを 3 種類実施した．テストはそれぞれ，学習範囲の問題 11 問を 10 分間，学習範囲から転移が可能な問題 9 問を 10 分間, 学習範囲とは異なる範囲の問題 4 問を 5 分間で行った．テストを行う際には，各問で関数化など構造化を行い他の問題やテストで再利用するよう指示を与えた．これらの評価は手順が正しいかを評価した手順点と，関数化など構造化を行い他の問題で再利用できているかを評価した構造化点の 2 種類にて 1 問 1 点として評価した．

事前テストの後，学習教材として FB 有りの既存システムを使う FB 有群 8 名と，既存システムから FB 機能を削除したシステムを使う FB 無群 7 名に分かれて 30 分間学習を行った．

事後では事前と同じテスト, 評価基準で評価した．

### 3.2 テスト結果と考察

手順点と構造化点で評価した 3 種類のテストの合計点の平均点と効果量を表 1 に示す．表 1 から両群ともに手順点，構造化点が向上していることがわかる．よって，より詳細に効果を確かめるために分散分析を行った結果から手順点，構造化点共に両群間に有意差はなく事前事後でどちらも有意に向上していることが分かった（$p<0.005$）．この結果から，提案手法が有効に働いていると考えられる．

また，FB の有無についての影響を調べるために Cohen's d を用いて効果量を分析した．その結果，手順点，構造化点共に両群において効果量の目安が大となった．しかしながら両群を比較すると，FB 有群のほうが手順点においては 28 ポイント, 構造化点においては 48 ポイント高い結果となった. これにより, FB が有効に働いていることがわかった．

表 1 テスト 3 種の合計点における分析結果

| | 手順点 | | | 構造化点 | | |
|---|---|---|---|---|---|---|
| | 事前 | 事後 | 効果量 | 事前 | 事後 | 効果量 |
| FB 有群 | 5.25 | 10.00* | 1.44(大) | 1.00 | 5.13* | 1.47(大) |
| FB 無群 | 4.29 | 7.29* | 1.16(大) | 1.00 | 3.71* | 0.99(大) |

*: $p (< 0.005)$

## 4. おわりに

本研究ではこれまでにプログラムを部品として獲得するための提案手法及び支援システムの開発を行い，学習における一定の有効性を示してきた[2), (3)]．しかし，システムの効果が提案手法によるものかシステムのフィードバックによるものかは定かではなかった．そこで本稿では，フィードバックの有無によって学習効果に差が出るかを検証した．検証の結果，少なくとも実施した 3 種類のテストの合計点においては提案手法が有効に働いていることが示唆された．またフィードバックの有無においてはどちらも一定の効果が見られたが，フィードバックを有したシステムがより効果が高いことがわかった．今後は，問題と学習範囲の関係がテストごとに異なることからそれらを詳細に分析する必要があると考える．

謝辞

本研究の一部は科研費・基盤研究(C)(18H11586), 科研費・基盤研究(B)(17H01839)の助成による．

### 参考文献

(1) B. Shneiderman, R. Mayer: “Syntactic/semantic interactions in programmer behavior: A model and experimental results”, *Int. J. Parallel Program.*, vol. 8, no. 3, pp. 219–238 (1979)

(2) 古池謙人, 東本崇仁: “プログラムにおける構造的理解のための部品の段階的拡張手法の提案とそのシステムの開発”, 教育システム情報学会誌, vol. 35, no. 2, pp. 215–220 (2018)

(3) 古池謙人, 東本崇仁: “プログラムにおける構造的理解のための部品の段階的拡張手法の提案とそのシステムの開発”, *教育システム情報学会特集論文研究会研究報告= JSiSE Res. Rep.*, vol. 8, no. 8, pp. 211–218 (2017)